MANIPULATING DATA

In the present world data is an important resource in the existence of any organisation. It must therefore be managed and used properly to the advantage of the organisation

Data Governance

- This is a data management concept concerning the capability that enables an organisation to ensure and maintain high data quality thought its lifespan,
- It is a collection of processes, roles, policies, standards and metrics that ensures the effective and efficient use of data/information in enabling an organisation to achieve its goal.
- It is defined as a system for defining who within an organisation has authority and control over data assets and how those data assets may be used.
- It encompasses the people, processes and technologies required to manage the and protect the data

The key focus areas in data governance include:

- **Availability** – is concerned with information being available to the users when they need it. For it to be available, it must be produced at the right time and stored properly for easy retrieval.
- **Usability** – this is concerned with information being in a form that can be used the consumer. Information must be in the correct format and in a form that can be understood.
- **Consistency** – Information must be consistent. Consistent with the facts in the ground and also though produced by different users the format is also the same. This makes it easier for the users to pick out important issues without necessarily reading everything.
- **Data integrity** – This is about trustworthiness of the information. The users must be able to believe in the information given to them.
- **Data security** – information must be held securely to increase its integrity. Protective measures must be put in place from the time the information is produced up to the time it is destroyed by the organisation.

Data governance encompasses the

- **People** – these are individuals who create, store or access the information. These people must know how to handle information
- **Processes** – these are activities that transform information from one form to another. The processes must be evaluated to ensure they do not distort information
- **Information technology/Equipment** – these are the machines that holds or process the information. The equipment must be in a safe environment to eliminate leakage or destruction of information.

Importance of Data Governance

- Improves the quality of data within an organisation
- Increases consistency and confidence in decision making
- Minimises legal issues that may arise due to mismanagement of data
- Ensure data security
- Allow an organisation to have a common understanding about how data is produced, handled, processed and consumed by the organisation staff

MANIPULATING DATA

1. DATA INTEGRITY

If a company's data is altered or deleted, and you have no way of knowing how, when and by whom, it can have a major impact on data-driven business decisions. **Data integrity** is the **maintenance of**, and the **assurance of the accuracy** and **consistency** of data.
It results in **reliability** and **trustworthiness** of data throughout its lifecycle.
The overall intent of any data integrity technique is to ensure data is recorded exactly as intended and upon later retrieval, ensures the data is the same as it was when it was originally recorded. In short, **data integrity aims to prevent unintentional changes to information**.
Any unintended changes to data as the result of a storage, retrieval or processing operation, including malicious intent, unexpected hardware failure, and human error, is failure of data integrity. If the changes are the result of unauthorized access, it may also be a failure of data security. It may result from a simple loses of pictures to even catastrophic loss of human life in a life-critical system.

Types of Data Integrity

- Physical
- Logical

Physical integrity

It deals with challenges associated with correctly storing and fetching the data itself. Challenges with physical integrity may include

- Electromechanical faults
- Design flaws
- Material fatigue
- Corrosion
- Power outages
- Natural disasters
- Acts of war and terrorism
- Other special environmental hazards such as ionizing radiation, extreme temperatures, pressures and g-forces.

Ways of ensuring physical integrity includes

- Redundant hardware (two or more hardware doing the same thing)
- An uninterruptible power supply
- Radiation hardened chips
- Error-correcting memory
- Use of a clustered file system
- Using file systems that employ block level checksums such as zfs,
- Storage arrays that compute parity calculations such as exclusive
- Use a cryptographic hash function
- Having a watchdog timer on critical subsystems.

Physical integrity often makes extensive use of error detecting algorithms known as error-correcting codes. Human-induced data integrity errors are often detected

through the use of simpler checks and algorithms, such as the Damm algorithm or Luhn algorithm.

Logical Integrity

This type of integrity is concerned with the correctness or rationality of a piece of data, given a particular context. This includes topics such as referential integrity and entity integrity in a relational database or correctly ignoring impossible sensor data in robotic systems. These concerns involve ensuring that the data "makes sense" given its environment. Challenges include software bugs, design flaws, and human errors. Common methods of ensuring logical integrity include

- Check constraints
- foreign key constraints
- program assertions.

Both physical and logical integrity often share many common challenges such as human errors and design flaws, and both must appropriately deal with concurrent requests to record and retrieve data.

Databases

Data integrity contains guidelines for data retention, specifying or guaranteeing the length of time data can be retained in a particular database. To achieve data integrity, these rules are consistently and routinely applied to all data entering the system, and any relaxation of enforcement could cause errors in the data. Implementing checks on the data as close as possible to the source of input (such as human data entry), causes less erroneous data to enter the system. Strict enforcement of data integrity rules results in lower error rates, and time saved troubleshooting and tracing erroneous data and the errors it causes to algorithms.

Data integrity also includes rules defining the relations a piece of data can have, to other pieces of data, such as a **Student** record being allowed to link to **Performance**, but not to unrelated data such as **Assets**. Data integrity often includes checks and correction for invalid data, based on a predefined set of rules.

Types of integrity constraints

Data integrity is normally enforced in a database system by a series of integrity constraints or rules. Three types of integrity constraints are **entity integrity**, **referential integrity** and **domain integrity**.

1. **Entity integrity** concerns the concept of a **primary key**. Entity integrity is an integrity rule, which states that every table must have a primary key and that the column or columns chosen to be the primary key should be unique and not null.

2. **Referential integrity** is a property of data stating that all of its references are valid. In the context of relational databases, it requires that if a value of one attribute (column) of a relation (table) references a value of another attribute (either in the same or a different relation), then the referenced value must exist. (i.e. a record in one table must always have a related record in another table if the two tables are linked together). For referential integrity to hold in a relational database, any column in a base table that is declared a foreign key can only contain either null values or values from a parent table's primary key or a candidate key. In Ms Access this is achieved by selecting **Enforce Referential Integrity** when creating relations

3. **Domain integrity** specifies that all columns in a relational database must be declared upon a defined domain. The primary unit of data in the relational data model is the data item. Such data items are said to be non-decomposable or atomic. A domain is a set of values of the same type. Domains are therefore pools of values from which actual values appearing in the columns of a table are drawn.

Having a single, well-controlled, and well-defined data-integrity system increases

- stability (one centralized system performs all data integrity operations)
- performance (all data integrity operations are performed in the same tier as the consistency model)
- re-usability (all applications benefit from a single centralized data integrity system)
- Maintainability (one centralized system for all data integrity administration).

Data Dictionary

This is a file or a set of files that contains a database's metadata. The data dictionary contains records about other objects in the database, such as data ownership, data relationships to other objects, and other data.

In a relational database, the metadata in the data dictionary includes the following:

- Names of all **tables** in the database and their owners
- The data/**Attributes** going to be held
- The **type** of data, This can be numeric, text, currency, date
- Names of all **indexes** and the columns to which the tables in those indexes relate
- Constraints defined on tables, including **primary keys**, **foreign-key** and **relationships** to other tables

Example of a Data Dictionary

ENTITY DESCRIPTION	
SYSTEM	DATE
AUTHOR	PAGE
NAME: **STUDENT**	VERSION:

NARRATIVE

This table holds details of each student in the database

Key	Attribute				Comment
√	Student Number	Text	6	[900001 – 999999]	
	Student Name	Text	30		
	Data of Birth	Date			
	Gender	Text	1	[M or F]	

Data Validation

Data validation is the process of ensuring data is valid and accepted. It tries to minimise errors in data especially during data entry.
Data validation is done automatically by a software when data is being entered.
In Database, the designer selects and defines these validations, which are then used to check if the data item fulfils them when entering new data into the system.

Types of Validation rules

Validation type	Explanation	Example of Usage
Check digit	The last one or two digits in a code are used to check the other digits are correct	Bar code readers in supermarkets use check digits
Format check	Checks the data is in the right format	Kenya car number plate is in the format LLL 999L where L is any letter and 9 is any number
Length check	Checks the data isn't too short or too long	A password which needs to be six letters long
Lookup table	Looks up acceptable values in a table	There are only seven possible days of the week
Presence check	Checks that data has been entered into a field	In most databases a key field cannot be left blank
Range check	Checks that a value falls within the specified range	Number of hours worked must be less than 50 and more than 0

https://www.youtube.com/watch?v=AeVJy-ow2b0

https://www.youtube.com/watch?v=aOVN0v-HWcQ
https://www.howtoexcel.org/tips-and-tricks/11-awesome-examples-of-data-validation/

.

2. Data Duplication and Modification Anomalies

- Data redundancy is a data organization issue that allows the unnecessary duplication of data within your database. Data redundancy has the following effects to a database
    - **Wastage of space** – Sometime you lack a complete information for a record. Therefore some fields have data while others are null
    - **Difficulties in Editing/Maintenance** - A change or modification, to redundant data, requires that you make changes to multiple fields of a database. This is difficult and time consuming and it may results in inconsistency of data i.e. the same data but different format
    - **Affects data integrity** – because of data inconsistency, data loses its integrity. When you have different format of the same data you don't know which is the correct one

For example

SalesStaff						
EmployeeID	**SalesPerson**	**SalesOffice**	**OfficeNumber**	**Customer1**	**Customer2**	**Customer3**
1003	Mary Smith	Chicago	312-555-1212	Ford	GM	
1004	John Hunt	New York	212-555-1212	Dell	HP	Apple
1005	Martin Hap	Chicago	312-555-1212	Boeing		

Notice that for each **SalesPerson** we have listed both the **SalesOffice** and **OfficeNumber**. There is duplicate sales person data. Duplicated information presents two problems: **It increases storage and decrease performance and it becomes more difficult to maintain data changes**.

Consider if we move the **Chicago office** to **Evanston, IL**. To properly reflect this in our table, we need to update the entries for all the **SalesPersons** currently in Chicago. Our table is a small example, but you can see if it were larger, that potentially this could involve hundreds of updates.

These situations are modification anomalies. There are three modification anomalies that can occur:

a) Insert Anomaly

EmployeeID	**SalesPerson**	**SalesOffice**	**OfficeNumber**	**Customer1**	**Customer2**	**Customer3**
1003	Mary Smith	Chicago	312-555-1212	Ford	GM	
1004	John Hunt	New York	212-555-1212	Dell	HP	Apple
1005	Martin Hap	Chicago	312-555-1212	Boeing		
???	???	Atlanta	312-555-1212			

There are facts we cannot record until we know information for the entire row. In our example we cannot record a new sales office until we also know the sales person. Why? Because in order to create the record, we need provide a primary key. In our case this is the EmployeeID.

b) Update Anomaly

EmployeeID	SalesPerson	SalesOffice	OfficeNumber	Customer1	Customer2	Customer3
1003	Mary Smith	Chicago	312-555-1212	Ford	GM	
1004	John Hunt	New York	212-555-1212	Dell	HP	Apple
1005	Martin Hap	Chicago	312-555-1212	Boeing		

In this case we have the same information in several rows. For instance if the office number changes, then there are multiple updates that need to be made. If we don't update all rows, then inconsistencies appear.

c) Deletion Anomaly

EmployeeID	SalesPerson	SalesOffice	OfficeNumber	Customer1	Customer2	Customer3
1003	Mary Smith	Chicago	312-555-1212	Ford	GM	
~~1004~~	~~John Hunt~~	~~New York~~	~~212-555-1212~~	~~Dell~~	~~HP~~	~~Apple~~
1005	Martin Hap	Chicago	312-555-1212	Boeing		

Deletion of a row causes removal of more than one set of facts. For instance, if John Hunt retires, then deleting that row cause us to lose information about the New York office.

NORMALIZATION

- Database **normalization** is the process of structuring a relational in order to reduce data redundancy and improve data integrity. It was first proposed by **Edgar F. Codd** as an integral part of his relational model.
- Normalisation also simplifies the database design so that it achieves the optimal structure composed of atomic elements (i.e. elements that cannot be broken down into smaller parts).
- Normalisation is an important part of relational database design, as it helps with the speed, accuracy, and efficiency of the database.
- Normalisation entails organizing the columns (**attributes**) and tables (**relations**) of a database to ensure that their dependencies are properly enforced by database integrity constraints. It is accomplished by applying some formal rules either by a process of synthesis (creating a new database design) or decomposition (improving an existing database design).

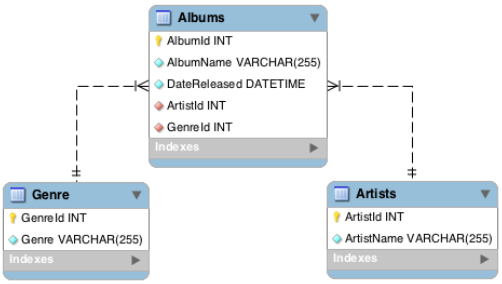
Advantages (Normalisation)

- Minimizes data redundancy (duplicate data).
- Minimizes null values.
- Results in a more compact database (due to less data redundancy/null values).
- Minimizes/avoids data modification issues.
- Simplifies queries.
- The database structure is cleaner and easier to understand. You can learn a lot about a relational database just by looking at its schema/structure.
- You can extend the database without necessarily impacting the existing data.
- Searching, sorting, and creating indexes can be faster, since tables are narrower, and more rows fit on a data page.

Example of a Normalized Database

When designing a relational database, one typically normalises the data before they create a schema. The database schema determines the organisation and the structure of the database – basically how the data will be stored.

Here's an example of a normalized database schema:



A basic schema diagram of a normalized database.

Levels of Normalization
There are various levels of normalisation, each one building on the previous level. These are

***1NF* (First Normal Form)**
A relation (table) is in 1NF if (and only if) the domain of each attribute contains only atomic (indivisible) values, and the value of each attribute contains only a single value from that domain.

***2NF* (Second Normal Form)**
A relation is in 2NF if it is in 1NF and every non-prime attribute of the relation is dependent on the whole of every candidate key.

***3NF* (Third Normal Form)**
A relation is in 3NF if it is in 2NF and every non-prime attribute of the relation is non-transitively dependent on every key of the relation.

3NF is strong enough to satisfy most applications, and the higher levels are rarely used, except in certain circumstances where the data (and its usage) requires it.

Example

First normal form (1NF)

As per the rule of first normal form, an attribute (column) of a table cannot hold multiple values. It should hold only atomic values.
Example: Suppose a company wants to store the names and contact details of its employees. It creates a table that looks like this:

emp_id	emp_name	emp_address	emp_mobile
101	Herschel	New Delhi	8912312390
102	Jon	Kanpur	8812121212 9900012222
103	Ron	Chennai	7778881212
104	Lester	Bangalore	9990000123 8123450987

Two employees (Jon & Lester) are having two mobile numbers so the company stored them in the same field as you can see in the table above.
This table is **not in 1NF** as the rule says "each attribute of a table must have atomic (single) values", the **emp_mobile** values for employees **Jon & Lester** violates that rule.
The name and address can also be split into atomic values
To make the table complies with 1NF we should have the data like this:

emp_id	emp_name	emp_address	emp_mobile
101	Herschel	New Delhi	8912312390
102	Jon	Kanpur	8812121212
102	Jon	Kanpur	9900012222
103	Ron	Chennai	7778881212
104	Lester	Bangalore	9990000123
104	Lester	Bangalore	8123450987

Second normal form (2NF)

A table is said to be in 2NF if:

- **Table is in 1NF (First normal form)**
- **No non-prime attribute is dependent on the proper subset of any candidate key of table**.

An attribute that is not part of any candidate key is known as non-prime attribute.
Example: Suppose a school wants to store the data of teachers and the subjects they teach. They create a table that looks like this: Since a teacher can teach more than one subjects, the table can have multiple rows for a same teacher.

teacher_id	subject	teacher_age
111	Maths	38
111	Physics	38
222	Biology	38
222	Chemistry	38
333	Physics	40
333	Chemistry	40

Candidate Keys: {teacher_id, subject}
Non prime attribute: teacher_age
The table is in 1 NF because each attribute has atomic values. However, it is not in 2NF because non prime attribute teacher_age is dependent on teacher_id alone which is a proper subset of candidate key. This violates the rule for 2NF as the rule says “**no** non-prime attribute is dependent on the proper subset of any candidate key of the table”.
To make the table complies with 2NF we can break it in two tables like this:

teacher_details table:

teacher_id	teacher_age
111	38
222	38
333	40

222	Biology
333	Physics
333	Chemistry

teacher_subject table:

teacher_id	subject
111	Maths
111	Physics

Now the tables comply with Second normal form (2NF).

Third Normal form (3NF)

A table design is said to be in 3NF if both the following conditions hold:

- Table must be in 2NF
- Transitive functional dependency of non-prime attribute on any super key should be removed.

An attribute that is not part of any candidate key is known as non-prime attribute.
In other words, 3NF can be explained like this: A table is in 3NF if it is in 2NF and for each functional dependency X-> Y at least one of the following conditions hold:

- X is a super key of table
- Y is a prime attribute of table

An attribute that is a part of one of the candidate keys is known as prime attribute.
Example: Suppose a company wants to store the complete address of each employee, they create a table named employee_details that looks like this:

emp_id	emp_name	emp_zip	emp_state	emp_city	emp_district
1001	John	282005	UP	Agra	Dayal Bagh
1002	Ajeet	222008	TN	Chennai	M-City
1006	Lora	282007	TN	Chennai	Urrapakkam
1101	Lilly	292008	UK	Pauri	Bhagwan
1201	Steve	222999	MP	Gwalior	Ratan

Super keys: {emp_id}, {emp_id, emp_name}, {emp_id, emp_name, emp_zip}…so on
Candidate Keys: {emp_id}
Non-prime attributes: all attributes except emp_id are non-prime as they are not part of any candidate keys.
Here, emp_state, emp_city & emp_district dependent on emp_zip. And, emp_zip is dependent on emp_id that makes non-prime attributes (emp_state, emp_city & emp_district) transitively dependent on super key (emp_id). This violates the rule of 3NF.
To make this table complies with 3NF we have to break the table into two tables to remove the transitive dependency:

employee table:

emp_id	emp_name	emp_zip
1001	John	282005
1002	Ajeet	222008
1006	Lora	282007
1101	Lilly	292008
1201	Steve	222999

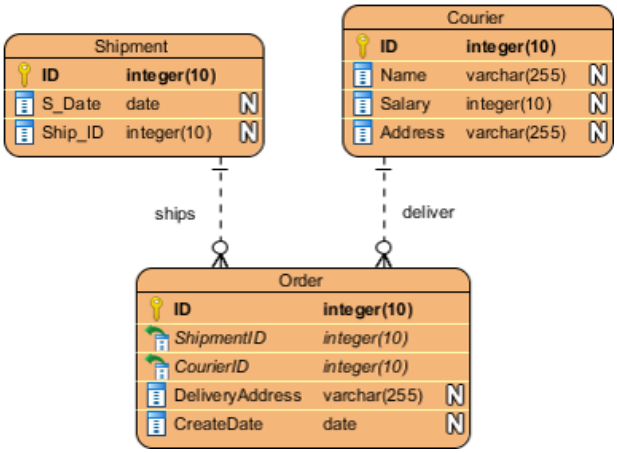
employee_zip table:

emp_zip	emp_state	emp_city	emp_district
282005	UP	Agra	Dayal Bagh
222008	TN	Chennai	M-City
282007	TN	Chennai	Urrapakkam
292008	UK	Pauri	Bhagwan
222999	MP	Gwalior	Ratan

Entity Relationship Diagram (ERD)

Database is absolutely an integral part of software systems.
An **ER Diagram** is a diagrammatical representation of entities and relations that exists between these entities.
An **entity** is defined as a **thing** capable of an independent existence that can be uniquely identified. It can be a physical thing e.g. student or an activity e.g. **Returning of Library books**
Entities can be thought of as **nouns**. Examples: a computer, an employee, a song, a mathematical theorem, etc.
A relationship captures how entities relate to one another. Relationships can be thought of as **verbs**, linking two or more nouns. Examples: an owns relationship between a company and a computer, a supervises relationship between an employee and a department.
Entities and relationships can both have attributes. Examples: an employee entity might have a Social Security Number (SSN) attribute, while a proved relationship may have a date attribute.
Every entity must have a uniquely identifying attributes called the entity's **primary key**.

Entity–relationship diagrams do not show single entities or single instances of relations. Rather, they show **entity sets** (all entities of the same entity type) and relationship sets (all relationships of the same relationship type). Examples: a particular song is an entity; the collection of all songs in a database is an entity set; the eaten relationship between a child and her lunch is a single relationship; the set of all such child-lunch relationships in a database is a relationship set.



Example of an E R diagram

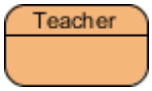
ERD Symbols

An ER Diagram contains **entities**, **attributes**, and **relationships**.

Entity

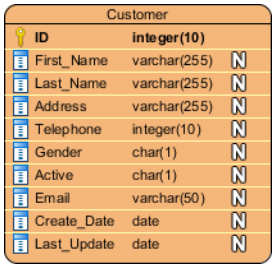
An ERD entity is a **definable thing or concept within a system**, such as a person/role (e.g. Student), object (e.g. Invoice), concept (e.g. Profile) or event (e.g. Transaction) (note: In ERD, the term "entity" is often used instead of "table", but they are the same).

In ER models, an entity is shown as a rounded rectangle, with its name on top and its attributes listed in the body of the entity shape.

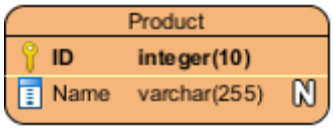


Entity Attributes

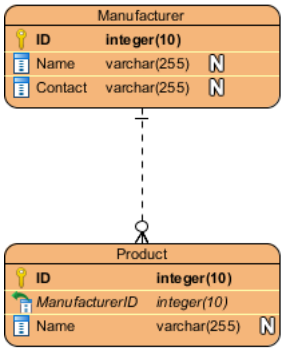
An attribute is a **property or characteristic of the** entity that holds it**.**
An attribute has a name that **describes the property** and a **type** that describes the kind of attribute it is, such as **varchar** for a **string**, and **int** for **integer**.
The ER diagram example below shows an entity with some attributes in it.



Primary Key
A primary key is a special kind of entity attribute that **uniquely defines a record in a database table**.



Foreign Key
A foreign key is an attribute that is a **reference to a primary key in a table**. It is a field present in another table and is used to create relationship between tables.



Relationship
A relationship between two entities signifies that the **two entities are associated with each other somehow**. For example, a student might enroll in a course. The entity Student is therefore related to Course, and a relationship is presented as a connector connecting between them.

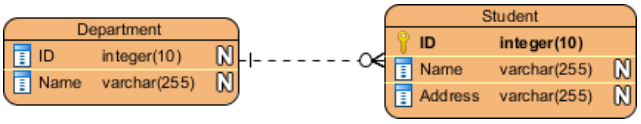
Cardinality
Cardinality defines the **possible number of occurrences in one entity which is associated with the number of occurrences in another**. For example, ONE team has MANY players. When present in an ERD, the entity Team and Player are inter-connected with a one-to-many relationship.
In an ER diagram, cardinality is represented as a crow's foot at the connector's ends. The three common cardinal relationships are one-to-one, one-to-many, and many-to-many.

- **One-to-One cardinality example**

Once occurrence of an instance of an entity in an entity set will have only a single occurrence of the same entity in another entity set. e.g.

- **One-to-Many cardinality example**

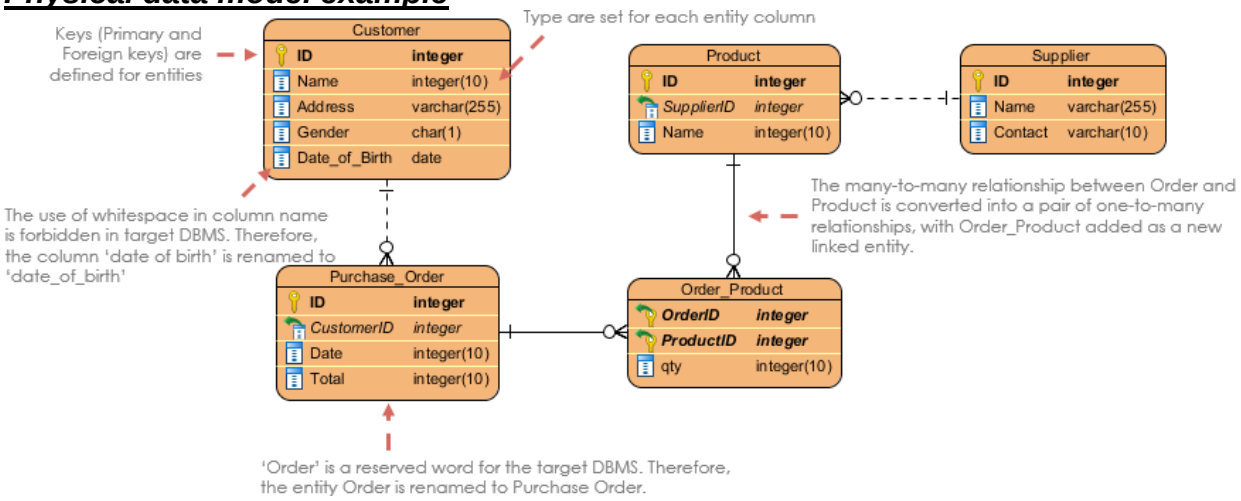
Once occurrence of an instance of an entity in an entity set will have several occurrences of the same entity in another entity set. e.g. A student can belong to one department but a department can have several students



- **Many-to-Many cardinality example**

Several occurrences of instances of entities in an entity set will have several occurrences of the same entities in another entity set. e.g. a teacher can teach more than one subject and a subject can be taught by several teachers.

Physical data model example


Keys (Primary and Foreign keys) are defined for entities
Type are set for each entity column
Customer
ID integer
Name integer(10)
Address varchar(255)
Gender char(1)
Date_of_Birth date
Product
ID integer
SupplierID integer
Name integer(10)
Supplier
ID integer
Name varchar(255)
Contact varchar(10)
The use of whitespace in column name is forbidden in target DBMS. Therefore, the column 'date of birth' is renamed to 'date_of_birth'
The many-to-many relationship between Order and Product is converted into a pair of one-to-many relationships, with Order_Product added as a new linked entity.
Purchase_Order
ID integer
CustomerID integer
Date integer(10)
Total integer(10)
Order_Product
OrderID integer
ProductID integer
qty integer(10)
'Order' is a reserved word for the target DBMS. Therefore, the entity Order is renamed to Purchase Order.

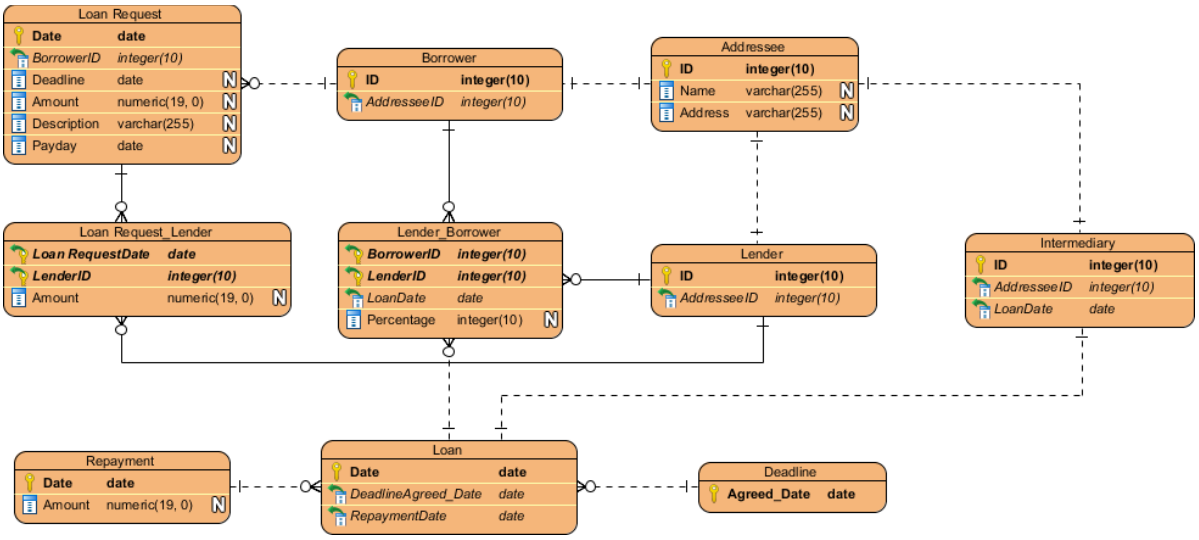
How to draw an ER diagram?

1. Draw the major entities involved in the scope.
2. Define the properties of entities by adding columns.
3. Review the ERD carefully and check if the entities and columns are enough to store the data of the system. If not, consider adding additional entities and columns. Usually, you can identify some transactional, operational and event entities in this step.

4. Consider the relationships between all entities and relate them with proper cardinality (e.g A one-to-many between entity Customer and Order). Don't worry if there are orphan entities. Although it's not common, it's legit.
5. Apply the technique of database normalization to re-structure the entities in a way that can reduce data redundancy and improve data integrity. For example, the details of the manufacturer might be stored under the Product entity initially. During the process of normalization, you may find that the detail keeps repeating records over records, then you can split it as a separate entity Manufacturer, and with a foreign key that links between Product and Manufacturer.

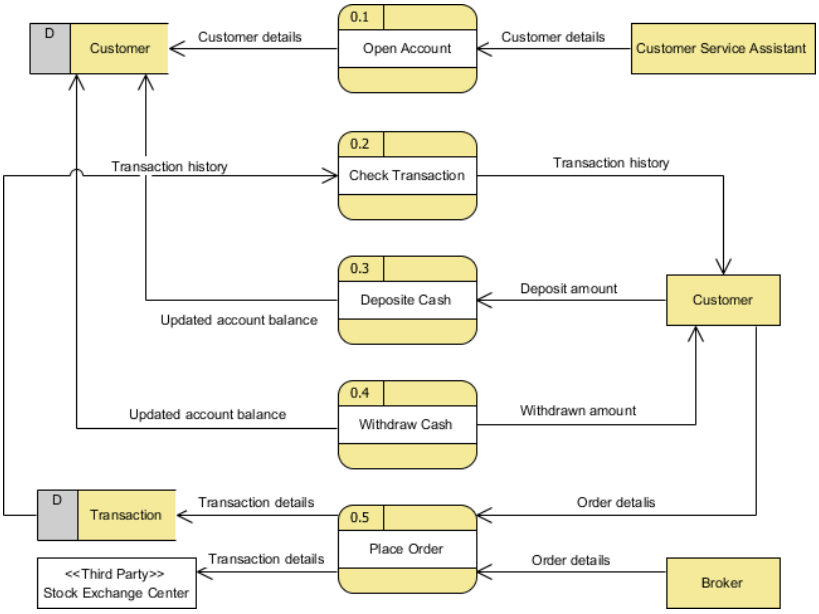
ER Diagram Example

Loan System

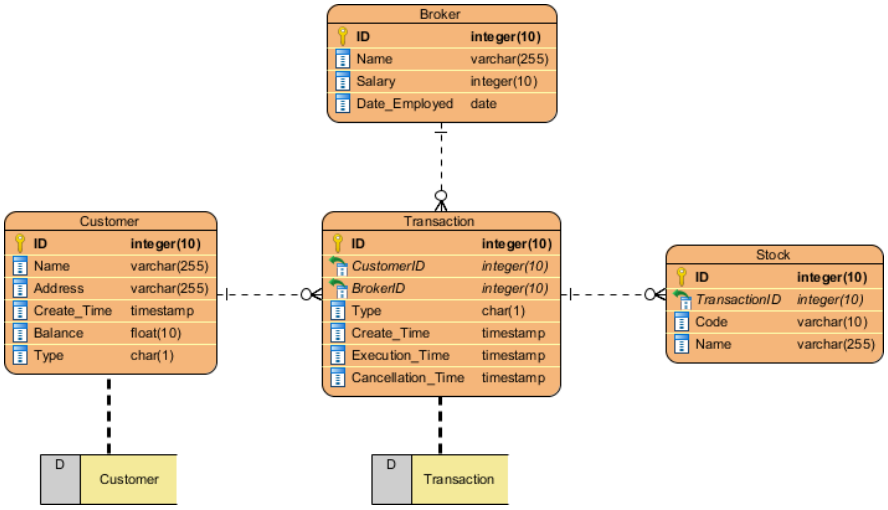


Using ERD with Data Flow Diagram (DFD)

In system analysis and design, Data Flow Diagram (DFD) can be drawn to visualize the flow of information within system processes. In a Data Flow Diagram, there is a symbol called Data Store, which represents a database table that provides the information needed by the system.



Since a physical ER Diagram provides a blueprint of an actual database, the entities in such an ERD are aligned with datastores in a DFD. You can draw ERD as a complement to DFD by representing the structure of information that flows within a system, or, on the contrary, to draw DFD in complementing an ERD by showing how the data will be utilized by the system in runtime.



Further Reading

- https://www.guru99.com/database-normalization.html
- https://www.sqa.org.uk/e-learning/DDFun01CD/page_09.htm
- https://www.sqa.org.uk/e-learning/MDBS01CD/page_26.htm
- https://www.youtube.com/watch?v=xoTyrdT9SZI – anomalies
- https://www.studytonight.com/dbms/database-normalization.php
- https://www.essentialsql.com/get-ready-to-learn-sql-database-normalization-explained-in-simple-english/
- https://www.visual-paradigm.com/guide/data-modeling/what-is-entity-relationship-diagram/#erd-data-

BIG DATA

INTRODUCTION

Data refers to the quantities, characters, or symbols on which operations are performed by a computer, which may be stored and transmitted in the form of electrical signals and recorded on magnetic, optical, or mechanical recording media. **Data** is used interchangeably with **information**

Big Data is a term used to describe a collection of data that is huge in size and yet growing exponentially with time. In it is data that is so large and complex and cannot be stored or processed efficiently by the traditional data management systems.

Examples of Big Data include:

- The **New York Stock Exchange** generates about ***one terabyte*** of new trade data per day.
- **Social Media**
  The statistic shows that ***500+terabytes*** of new data is generated in terms of photos and videos uploads, text messages, comments etc
- A single **Jet engine** can generate ***10+terabytes*** of data in ***30 minutes*** of flight time. With many thousand flights per day, generation of data reaches up to many ***Petabytes.***

Types:

1. **Structured**
2. **Unstructured**
3. **Semi-structured**

1. Structured

This is data stored, accessed and processed in the form of fixed format. It includes data stored in data management systems

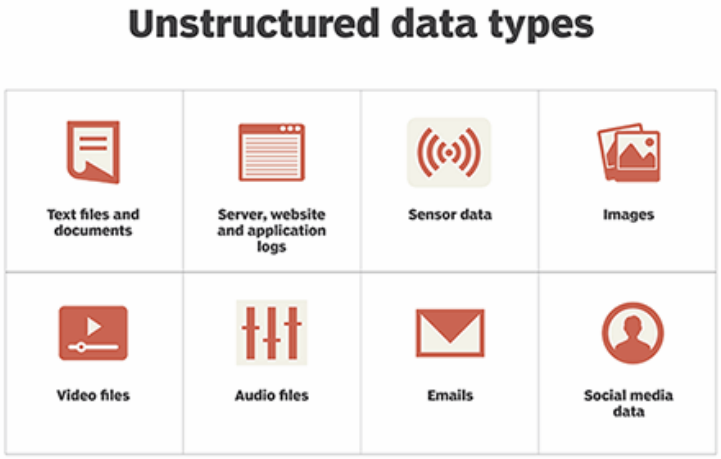
Examples of Structured Data

An 'Employee' table in a database is an example of Structured Data

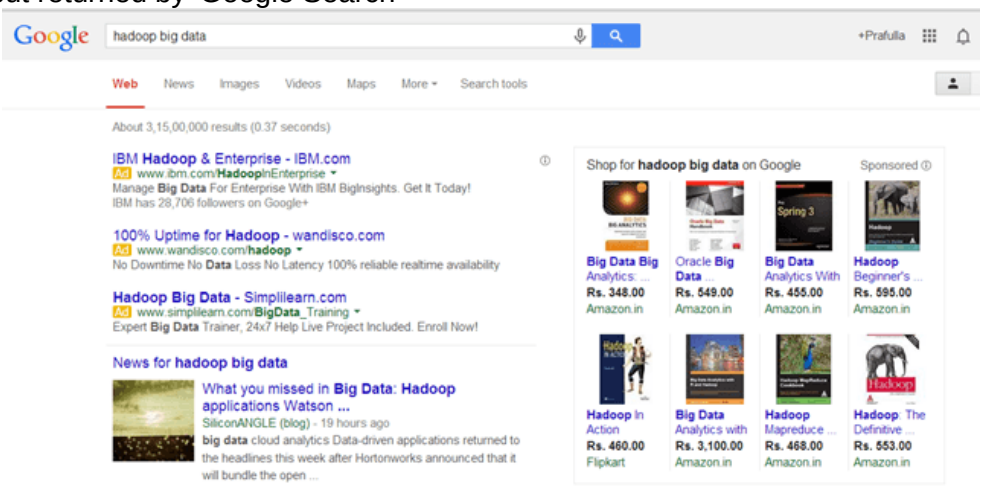
Employee_ID	Employee_Name	Gender	Department	Salary_In_lacs
2365	Rajesh Kulkarni	Male	Finance	650000
3398	Pratibha Joshi	Female	Admin	650000
7465	Shushil Roy	Male	Admin	500000
7500	Shubhojit Das	Male	Finance	500000
7699	Priya Sane	Female	Finance	550000

2. Unstructured

Data lacking form or structures.
It is a heterogeneous data source containing a combination of simple text files, images, videos etc.
Unstructured data needs to be organised in a structured way before it can be processed



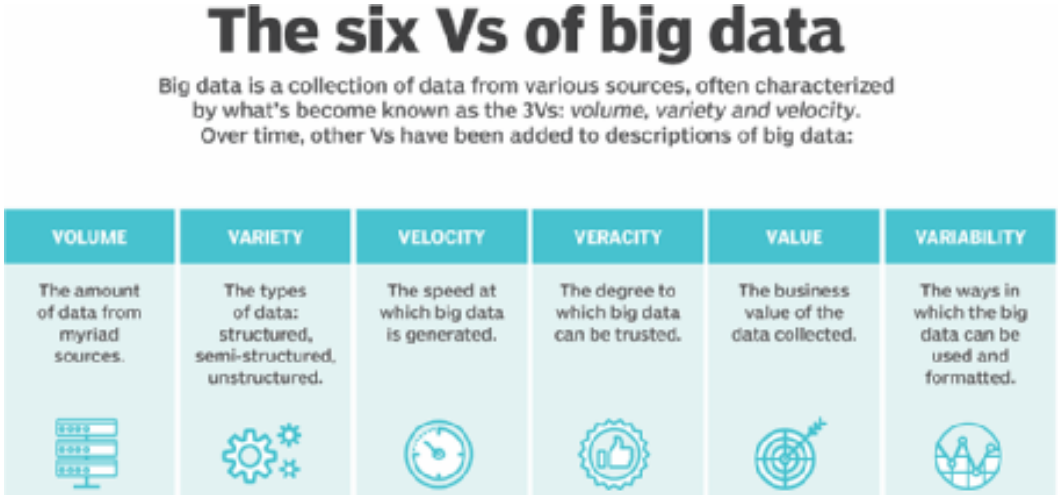
Examples of Un-structured Data
The output returned by 'Google Search'



3. Semi-structured

Semi-structured data can contain both the forms of data. We can see semi-structured data as a structured in form but it is actually not defined with e.g. a table definition in relational DBMS. Example of semi-structured data is a data represented in an XML file.
Examples Of Semi-structured Data
Personal data stored in an XML file
<rec><name>Prashant Rao</name><sex>Male</sex><age>35</age></rec>
<rec><name>Seema R.</name><sex>Female</sex><age>41</age></rec>
<rec><name>Satish Mane</name><sex>Male</sex><age>29</age></rec>
<rec><name>Subrato Roy</name><sex>Male</sex><age>26</age></rec>

Characteristics (Big Data)


The six Vs of big data
Big data is a collection of data from various sources, often characterized by what's become known as the 3Vs: volume, variety and velocity. Over time, other Vs have been added to descriptions of big data:
VOLUME
The amount of data from myriad sources.
VARIETY
The types of data: structured, semi-structured, unstructured.
VELOCITY
The speed at which big data is generated.
VERACITY
The degree to which big data can be trusted.
VALUE
The business value of the data collected.
VARIABILITY
The ways in which the big data can be used and formatted.

(i) Volume –

- Volume refers to the unimaginable amounts of information generated every second
- The data is generated from social media, cell phones, cars, credit cards, M2M sensors, images and video. The name Big Data itself is related to a size, which is enormous.

(ii) Variety –

- It refers to heterogeneous (different) sources/format and the nature of data, both structured and unstructured.
- The data in the form of emails, photos, videos, monitoring devices, PDFs, audio. This variety of unstructured data poses certain issues for storage, mining and analysing data.
- Big data also encompasses a wide variety of data types, including structured data in SQL databases and data warehouses, unstructured data, such as text and document files, and semi-structured data, such as web server logs or streaming data from sensors..

(iii) Velocity –

- This refers to the speed of generation of data. How fast the data is generated and processed to meet the demands and/or determines real potential in the data.
- Sets of big data are updated on a real- or near-real-time basis, compared with daily, weekly or monthly updates in many traditional data warehouses.
- Big Data Velocity deals with the speed at which data flows in from sources like business processes, application logs, networks, and social media sites, sensors, Mobile devices, etc. The flow of data is massive and continuous.

(iv) Variability –

- This refers to the inconsistency which can be shown by the data at times, thus hampering the process of being able to handle and manage the data effectively.

(v) Veracity –

- It refers to the degree of certainty in the data sets (i.e. how reliable is the data. Costs of finding errors v costs of accepting errors, Legal consequences).
- Uncertain raw data collected from multiple sources, such as social media platforms and webpages, can cause serious data quality issues that may be difficult to pinpoint. For example, a company that collects data from hundreds of sources may be able to identify inaccurate data, but its analysts need data lineage information to trace where the data is stored so they can correct the issues.

Value:

- It is not just the amount of data that we store or process that matters but the amount of valuable, reliable and trustworthy data that needs to be stored, processed, analysed to find insights.

https://searchdatamanagement.techtarget.com/definition/big-data

Benefits of Big Data Processing

Ability to process Big Data brings in multiple benefits, such as-

- **Businesses can utilize outside intelligence while taking decisions**

Access to social data from search engines and sites like facebook, twitter are enabling organisations to fine tune their business strategies. Online loaning apps utilise Facebook and WhatsApp data to determine how much to give to an individual

- **Improved customer service**

Traditional customer feedback systems are getting replaced by new systems designed with Big Data technologies. In these new systems, Big Data and natural language processing technologies are being used to read and evaluate consumer responses.

- **Early identification of risk to the product/services, if any**
- **Better operational efficiency**

Big Data technologies can be used for creating a staging area or landing zone for new data before identifying what data should be moved to the data warehouse. In addition, such integration of Big Data technologies and data warehouse helps an organisation to offload infrequently accessed data.

INFRASTRUCTURE AND SERVICES: BIG DATA

Infrastructure is the cornerstone of Big Data architecture. Possessing the right tools for collecting, storing, processing and analysing your data is crucial in any Big Data project.

1. Collection Methods

Transactional Data

- Transactional data includes multiple variables, such as what, how much, how and when customers purchased as well as what promotions or coupons they used.
- Point of Sale (POS) software can automatically store this information in a CRM (Customer Relationship Management) software.
- This data can also be sold by supermarkets and big stores to users who require it

Online Marketing Analytics

- They collect data when a user browse a website
    - Google Analytics has the ability to provide a lot of demographic insight on each visitor. This information is useful is building marketing campaigns, as well as website performance analysis.
    - Heatmaps provide information on which sections of each website page generate the most ‘action’ (mouse clicks or interactions).
    - Social media analytics allows for customer demographic as well as behavioural analysis. And powerful Facebook marketing tools can help you market to audiences that mirror your current following.

Social Media

- Social media is used in many ways on a frequent basis: networking, procrastinating, gossiping, sharing, educating, games etc.
- 71 percent of internet users are social network users and by next year, it is estimated that there will be around 2.77 billion social media users around the globe. That equates to a LOT of social media data gathered!

Loyalty Cards

- The loyalty cards system is great as it rewards repeat customers and encourages more shopping.
- There are so many businesses willing to give customers a discount simply in exchange for their personal information. This is because this data is valuable to them and they can predict many shopping habits and in turn develop customised marketing efforts targeting those customers.
- Loyalty programs have the power to double overall sales by encouraging repeat shopping.

Maps

- Maps information has the potential to provide businesses with customer location demographics and a detailed picture of the kind of people who live and work in certain areas.

2. Database Systems

Hadoop

Hadoop is essentially an open-source framework for processing, storing and analysing data. The fundamental principle behind Hadoop is rather than tackling one monolithic block of data all in one go, it's more efficient to break up & distribute data into many parts, allowing processing and analysing of different parts concurrently.

When hearing Hadoop discussed, it's easy to think of Hadoop as one vast entity; this is a myth. In reality, Hadoop is a whole ecosystem of different products, largely presided over by the Apache Software Foundation. Some key components include:

- **HDFS**- The default storage layer
- **MapReduce**- Executes a wide range of analytic functions by analysing datasets in parallel before 'reducing' the results. The "Map" job distributes a query to different nodes, and the "Reduce" gathers the results and resolves them into a single value.
- **YARN**- Responsible for cluster management and scheduling user applications
- **Spark**- Used on top of HDFS, and promises speeds up to 100 times faster than the two-step MapReduce function in certain applications. Allows data to loaded in-memory and queried repeatedly, making it particularly apt for machine learning algorithms

The main advantages of Hadoop are its

1. **Cost -** it's open source, it's free and available for anyone to use, and can run off cheap commodity hardware.
2. **Time** - it processes multiple 'parts' of the data set concurrently, making it a comparatively fast tool for retrospective, in-depth analysis.

Hadoop-on-Premium services such as Cloudera, Hortonworks and Splice offer the Hadoop framework with greater security and support, with added system & data management tools and enterprise capabilities.

NoSQL

NoSQL, which stands for **Not Only SQL,** is a term used to cover a range of different database technologies. NoSQL databases are adept at processing dynamic, semi-structured data with low latency, making them better tailored to a Big Data environment.

The different strengths and uses of Hadoop and NoSQL are often described as "operational" and "analytical". NoSQL is better suited for "operational" tasks; interactive workloads based on selective criteria where data can be processed in near real-time. Hadoop is better suited to high-throughput, in-depth analysis in retrospect, where the majority or all of the data is harnessed. Since they serve different purposes, Hadoop and NoSQL products are sometimes marketed concurrently. Some NoSQL databases, such as HBase, were primarily designed to work on top of Hadoop.

Some big names in NoSQL field include Apache Cassandra, MongoDB, and Oracle NoSQL. Many of the most widely used NoSQL technologies are open source, meaning security and troubleshooting may be an issue. It also places less focus on atomicity and consistency than on performance and scalability. Premium packages of NoSQL databases (such as Datastax for Cassandra) work to address these issues.

Massively Parallel Processing (MPP)

MPP technologies process massive amounts of data in parallel. Hundreds (or potentially even thousands) of processors, each with their own operating system and memory, work on different parts of the same programme.

MPP usually runs on expensive data warehouse appliances, whereas Hadoop is most often run on cheap commodity hardware (allowing for inexpensive horizontal scale out). MPP uses SQL, and Hadoop uses Java as default. As with all technologies in this article, MPP has crossovers with the other technologies, Teradata, an MPP technology, has an ongoing partnership with Hortonworks (a Hadoop-on-Premium service).

Many of the major players in the MPP market have been acquired by technology vendor behemoths; Netezza, for instance, is owned by IBM, Vertica is owned by HP and Greenplum is owned by EMC.

Cloud

Cloud computing refers to a broad set of products that are sold as a service and delivered over a network. In other infrastructural approaches, when setting up your big architecture you need to buy hardware and software for each person involved with the processing and analysing of your data. In cloud computing, your analysts only require access to application- a web-based service where all of the necessary resources and programmes are hosted. In cloud computing, up-front costs are minimal as you typically only pay for what you use, and scale out from there.

Of course, having your data hosted by third party can raise questions about security; many choose to host their confidential information in-house, and use the cloud for less private data.

A lot of big names in IT offer cloud computing solutions; **Google** has a whole host of Cloud computing products, including BigQuery, specifically designed for the processing and management of Big Data; **Amazon Web Services** also has a wide range, included EMR for Hadoop, RDS for MySQL and DynamoDB for NoSQL. There are also vendors such as **Infochimps** and **Mortar** specifically dedicated to offering cloud computing solutions.

3. Data Storage

Data Warehousing/Enterprise Data Warehouse/ Decision Support Database

- This is **a technique for collecting and managing data from varied sources to provide meaningful business insights**. It is a blend of technologies and components, which aids the strategic use of data.
- The data is used for reporting and data analysis not transaction processing.
- The data structure and schema are defined in advance to optimise for fast SQL queries.
- When data is acquired, it is cleaned, enriched, and transformed so it can act as the “single source of truth” that users can trust
- The decision support database (Data Warehouse) is maintained separately from the organisation's operational database.
- It provides users with current and historical decision support information which is difficult to access or present in the traditional operational data store.
- It merges information coming from different sources into one comprehensive database

What Is a Data Warehouse Used For?

Here, are most common sectors where Data warehouse is used:

Airline:

- In the Airline system, it is used for operation purpose like crew assignment, analyses of route profitability, frequent flyer program promotions, etc.

Banking:

- It is widely used in the banking sector to manage the resources available on desk effectively. Few banks also use it for the market research, performance analysis of the product and operations.

Healthcare:

- Healthcare sector also used Data warehouse to strategize and predict outcomes, generate patient's treatment reports, share data with tie-in insurance companies, medical aid services, etc.

Public sector:

- In the public sector, data warehouse is used for intelligence gathering. It helps government agencies to maintain and analyse tax records, health policy records, for every individual.

Investment and Insurance sector:

- In this sector, the warehouses are primarily used to analyse data patterns, customer trends, and to track market movements.

Retain chain:

- In retail chains, Data warehouse is widely used for distribution and marketing. It also helps to track items, customer buying pattern, promotions and used for determining pricing policy.

Telecommunication:

- A data warehouse is used in this sector for product promotions, sales decisions and to make distribution decisions.

Hospitality Industry:

- This Industry utilises warehouse services to design as well as estimate their advertising and promotion campaigns where they want to target clients based on their feedback and travel patterns.

Advantages & Disadvantages

Advantages of Data Warehouse:

- Data warehouse allows business users to quickly access critical data from some sources all in one place.
- Data warehouse provides consistent information on various cross-functional activities. It is also supporting ad-hoc reporting and query.
- Data Warehouse helps to integrate many sources of data to reduce stress on the production system.
- Data warehouse helps to reduce total turnaround time for analysis and reporting.
- Restructuring and Integration make it easier for the user to use for reporting and analysis.
- Data warehouse allows users to access critical data from the number of sources in a single place. Therefore, it saves user's time of retrieving data from multiple sources.
- Data warehouse stores a large amount of historical data. This helps users to analyse different time periods and trends to make future predictions.

Disadvantages of Data Warehouse:

1. Not an ideal option for unstructured data.
2. Creation and Implementation of Data Warehouse is surely time confusing affair.
3. Data Warehouse can be outdated relatively quickly
4. Difficult to make changes in data types and ranges, data source schema, indexes, and queries.
5. The data warehouse may seem easy, but actually, it is too complex for the average users.

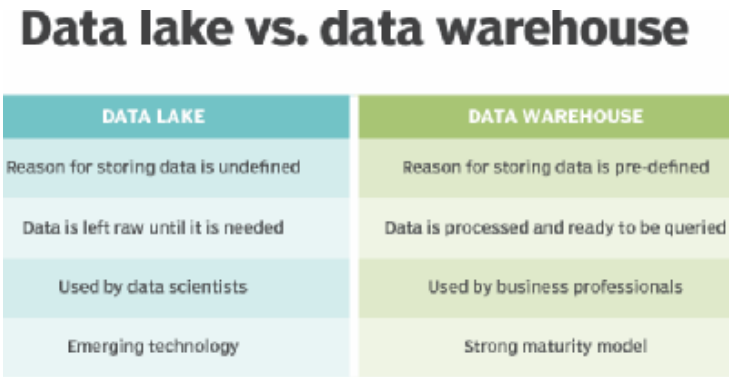
6. Organisations need to spend lots of their resources for training and Implementation purpose.

Data Lake

- This is a centralised repository that holds a vast amount of raw data in its **native format** until it is needed. Data is stored as-is, without having to first structure the data
- It stores relational data from line of business applications, and non-relational data from mobile apps, IoT devices, and social media.
- The structure of the data or schema is not defined when data is captured. This means you can store all of your data without careful design or the need to know what questions you might need answers for in the future.
- A data lake uses a flat architecture to store data. Each data element in a lake is assigned a unique identifier and tagged with a set of extended metadata tags. When a business question arises, the data lake can be queried for relevant data, and that smaller set of data can then be analysed to help answer the question.
- A data lake can be established within an organization's data centres or using cloud services from vendors such as Amazon, Google and Microsoft.

Data Lake vs. Data Warehouse

Characteristics	Data Warehouse	Data Lake
Data	Relational from transactional systems, operational databases, and line of business applications	Non-relational and relational from IoT devices, web sites, mobile apps, social media, and corporate applications
Schema	Designed prior to the DW implementation (schema-on-write)	Written at the time of analysis (schema-on-read)
Price/ Performance	Fastest query results using higher cost storage	Query results getting faster using low-cost storage
Data Quality	Highly curated data that serves as the central version of the truth	Any data that may or may not be curated (i.e. raw data)
Users	Business analysts	Data scientists, Data developers, and Business analysts (using curated data)
Analytics	Batch reporting, BI and visualizations	Machine Learning, Predictive analytics, data discovery and profiling


Data lake vs. data warehouse
DATA LAKE
DATA WAREHOUSE
Reason for storing data is undefined
Reason for storing data is pre-defined
Data is left raw until it is needed
Data is processed and ready to be queried
Used by data scientists
Used by business professionals
Emerging technology
Strong maturity model

- https://aws.amazon.com/big-data/datalakes-and-analytics/what-is-a-data-lake/
- https://www.talend.com/resources/data-lake-vs-data-warehouse/

Distributed/Cloud Based Storage system

- This is a model of computer data storage where digital data is stored to an off-site storage system maintained by a third party.
- The physical storage consists of several servers in different locations owned by an organisation or a service provider.

- The service provider may have several location allowing for duplication of data for the purpose of security/back up and faster access.
- The cloud storage provider manage data availability and accessibility. They also ensures contusive environment for the machines
- The facilities that house cloud storage system are called **data centres**
- Benefits of cloud storage include:
    - Accessing the data from any location
    - Reduce the cost of investing in storage devices or servers

https://www.villanovau.com/resources/bi/6-ways-companies-can-collect-your-data/

4. Big Data Transmission

- The default mechanism for connecting from on-premises infrastructure to cloud data centres is via IP network.
- The traditional methods of sending and sharing data over IP networks rely on a networking protocol known as TCP and other protocols such as HTTP and FTP.
- These work well for sending relatively a small data set over the internet (up to 20Gbps).
- Transmission of huge data files is increasingly becoming a common place.
- Real time transmission requires low **latency** and sufficient **throughput.**
- Organisations using the data require **minimal delays** and **reliable throughput** to avoid interruption of service.
- However, the **reliability** and **congestion control mechanisms** of TCP do not deal well with high latency and packet loss on long distance networks thereby causing transfer to become impractically slow
- A common approach to solving throughput problem is to purchase more bandwidth. But a bigger pipe does not address the underlying limitations of TCP that result in extremely inefficient utilisation of bandwidth in high latency scenarios
- The solution is to development and use of other protocols to replace or enhance TCP. The new protocols that are being used include Reliable Multi-Destination Transport (RMDT), UDP-based Data Transfer Protocol (UDT), Quick UDP Internet Connections (QUIC)), and advances in fibre optics, advances in data compression. RMDT is a fast data transmission protocol over high capacity WAN networks. It easily handles high packets delays and jitters as well as significant packet losses on a channel. The protocol does not require special hardware units.
- UDP-UDT is a high performance data transfer protocol designed specifically for the high volume transfer of large datasets over high speed WAN

IMPACT OF STORING BIG DATA

Although big data has an impact on the way organisation do their activities, it also introduces some concerns. These include

- **Access**

- The data is stored off-site location away from the enterprise. An enterprise must make a decision of who they are going to use to store the data. As much as there are several storage sites. One must select one that will fulfils their requirements.
- Secondly, an organisation must decide who sees this organisation's data. If outsiders at to access this data then the organisations must be able to allow this to happen. Even within the organisation, there must be a decision of who is going to see what
- To be able to access this data one must select a query language to do that. types of query language used,

- **Transmission time;**

- Different media transmit data at different speeds. An organisation must select the best media that will allow for faster and effect transfer of data.

- Bandwidth refers to the amount of data that can be moved from one point to another. The bigger the bandwidth the better. However, this introduces another problem: Cost. A compromise should be struck between bandwidth and cost that ensures that data can be moved from an off-site storage to an organisation within the reasonable time,
- requirements for building/upgrading networks and systems to cope with increasing data movement,
- international aspects,
- latency affecting processing at different locations especially affecting real time processing,
- synchronisation problems

- **Security;**

- theft of data from online/third party storage,
- ransomware,
- DDoS attacks,
- insider attacks,
- data misuse by authorised users,
- insertion of fake data,
- changes to data at source level such as changes to data or metadata formats at source can cause data to go to the wrong place,
- access control
- security audits
- physical security.

- **Processing time;**

- balance between security - encrypting all data - and processing time
- problems with having to decrypt-process-encrypt
- process types (e.g. batch, real time, stream)
- processing architecture (lambda and kappa)
- software
- problems with data noise and corruption

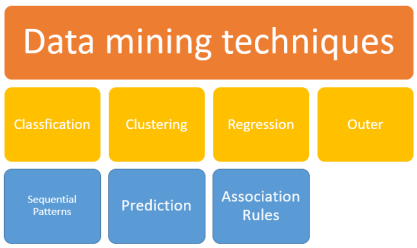
https://www.import.io/post/all-the-best-big-data-tools-and-how-to-use-them/

12.3.4 MANIPULATING THE BIG DATA

1. Data Mining

- **Def 1:** Data mining is looking for hidden, valid, and potentially useful patterns in huge data sets. Data Mining is all about discovering unsuspected/ previously unknown relationships amongst the data.
- **Def 2:** Data mining is the process of looking at large banks of information to generate new information. Data mining does not create new data. Instead, data mining is about extrapolating patterns and new knowledge from the data you've already collected.
- It is a multi-disciplinary skill that uses **machine learning**, **statistics**, **Artificial Intelligence** and **database technology**.
- The insights derived via Data Mining can be used for marketing, fraud detection, and scientific discovery, etc.

Data Mining Techniques



a) Classification:

This analysis is used to retrieve important and relevant information about data, and metadata. This data mining method helps to classify data in different classes.

b) Clustering:

Clustering analysis is a data mining technique to identify data that are like each other. This process helps to understand the differences and similarities between the data.

c) Regression:

Regression analysis is the data mining method of identifying and analysing the relationship between variables. It is used to identify the likelihood of a specific variable, given the presence of other variables.

d) Association Rules:

This data mining technique helps to find the association between two or more Items. It discovers a hidden pattern in the data set.

e) Outer detection:

This type of data mining technique refers to observation of data items in the dataset which do not match an expected pattern or expected behaviour. This technique can be used in a variety of domains, such as intrusion, detection, fraud or fault detection, etc. Outer detection is also called Outlier Analysis or Outlier mining.

f) Sequential Patterns:

This data mining technique helps to discover or identify similar patterns or trends in transaction data for certain period.

g) Prediction:

Prediction has used a combination of the other data mining techniques like trends, sequential patterns, clustering, classification, etc. It analyses past events or instances in a right sequence for predicting a future event.

https://www.datasciencecentral.com/profiles/blogs/the-7-most-important-data-mining-techniques

Challenges of Implementation of Data mine:

- Skilled Experts are needed to formulate the data mining queries.
- Overfitting: Due to small size training database, a model may not fit future states.
- Data mining needs large databases which sometimes are difficult to manage
- Business practices may need to be modified to determine to use the information uncovered.
- If the data set is not diverse, data mining results may not be accurate.
- Integration information needed from heterogeneous databases and global information systems could be complex

Benefits of Data Mining:

- Data mining technique helps companies to get knowledge-based information.
- Data mining helps organisations to make the profitable adjustments in operation and production.
- The data mining is a cost-effective and efficient solution compared to other statistical data applications.
- Data mining helps with the decision-making process.
- Facilitates automated prediction of trends and behaviours as well as automated discovery of hidden patterns.
- It can be implemented in new systems as well as existing platforms
- It is the speedy process, which makes it easy for the users to analyse huge amount of data in less time.

Disadvantages of Data Mining

- There are chances of companies may sell useful information of their customers to other companies for money. For example, American Express has sold credit card purchases of their customers to the other companies.
- Many data mining analytics software is difficult to operate and requires advance training to work on.
- Different data mining tools work in different manners due to different algorithms employed in their design. Therefore, the selection of correct data mining tool is a very difficult task.
- The data mining techniques are not accurate, and so it can cause serious consequences in certain conditions.

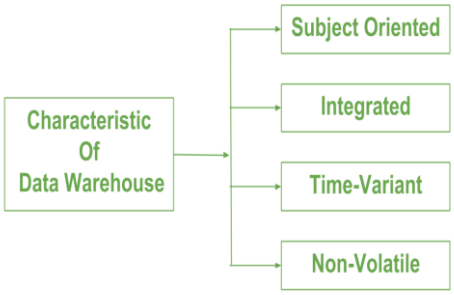
Data Mining Applications

Applications	Usage
Communications	Data mining techniques are used in communication sector to predict customer behaviour to offer highly targeted and relevant campaigns.
Insurance	Data mining helps insurance companies to price their products profitable and promote new offers to their new or existing customers.
Education	Data mining benefits educators to access student data, predict achievement levels and find students or groups of students which need extra attention. For example, students who are weak in maths subject
Manufacturing	With the help of Data Mining Manufacturers can predict wear and tear of production assets. They can anticipate maintenance which helps them reduce them to minimize downtime.
Banking	Data mining helps finance sector to get a view of market risks and manage regulatory compliance. It helps banks to identify probable defaulters to decide whether to issue credit cards, loans, etc.
Retail	Data Mining techniques help retail malls and grocery stores identify and arrange most sellable items in the most attentive positions. It helps store owners to comes up with the offer which encourages customers to increase their spending.
Service Providers	Service providers like mobile phone and utility industries use Data Mining to predict the reasons when a customer leaves their company. They analyse billing details, customer service interactions, complaints made to the company to assign each customer a probability score and offers incentives.
E-Commerce	E-commerce websites use Data Mining to offer cross-sells and up-sells through their websites. One of the most famous names is Amazon, who use Data mining techniques to get more customers into their e-Commerce store.
Super Markets	Data Mining allows supermarkets develop rules to predict if their shoppers were likely to be expecting. By evaluating their buying pattern, they could find woman customers who are most likely pregnant. They can start targeting products like baby powder, baby shop, diapers and so on.
Crime Investigation	Data Mining helps crime investigation agencies to deploy police workforce (where is a crime most likely to happen and when?), who to search at a border crossing etc.
Bioinformatics	Data Mining helps to mine biological data from massive datasets gathered in biology and medicine.

2. Data Warehousing

- A data warehouse is a relational or multidimensional database that is designed for query and analysis. It provides data that can be used to make strategic decision.
- Data warehouses are not created for transaction processing
- Data warehouses usually consolidate historical and analytic data collected from different sources.
- A data warehouse usually stores many months or years of data to support historical analysis..
- Users of the data warehouse perform data analyses that are often time-related. Examples include consolidation of last year's sales figures, inventory analysis, and profit by product and by customer. More sophisticated analyses include trend analyses and data mining, which use existing data to forecast trends or predict futures. The data warehouse typically provides the foundation for a business intelligence environment.

Characteristics of Data Warehouse



1. **Subject-oriented –**
   - A data warehouse is always a subject oriented i.e. it delivers information about a theme instead of organisation's current operations.
   - That means the data warehousing process is proposed to handle with a specific theme which is more defined.
   - These themes can be sales, distributions, marketing etc.

2. **Integrated**
   - The way data is extracted and transformed is uniform, regardless of the original source. This is because the structure of the data storage is defined before the data is stored.
   - The data also has reliable naming conventions, format and codes. Integration of data warehouse benefits in effective analysis of data.

3. **Time-Variant**
   - Data is organised via time-periods (weekly, monthly, annually, etc.).
   - The data resided in data warehouse is predictable with a specific interval of time and delivers information from the historical perspective.
   - Another feature of time-variance is that once data is stored in the data warehouse then it cannot be modified, alter, or updated.

4. **Non-Volatile**

- The data that resided in data warehouse is permanent. The data cannot be erased or deleted when new data is inserted.
- The data is read-only and refreshed at particular intervals. Two types of data operations done in the data warehouse are:
  - Data Loading
  - Data Access

https://dzone.com/articles/data-warehouse-characteristics-and-benefits
https://www.geeksforgeeks.org/characteristics-and-functions-of-data-warehouse/

Functions of a Data Warehouse

- **Data Extraction** − Involves gathering data from multiple heterogeneous/different sources. The data is related to or can be used by the organisation in decision making.
- **Data Cleaning** − Involves finding and correcting the errors in data.
- **Data Transformation** − Involves converting the data from legacy format to warehouse format.
- **Data Loading** − Involves sorting, summarizing, consolidating, checking integrity, and building indices and partitions.
- **Refreshing** − Involves updating from data sources to warehouse.

https://www.tutorialspoint.com/dwh/dwh_quick_guide.htm

Data Warehouse Benefits?

1. The Enablement of Better Decision-Making

- The organisation's strategic decision-makers no longer have to make important business decisions based on partial or limited data.
- They are now backed up by facts and statistics housed within data warehouses that can be recalled at any time.

2. Quick and Easy Data Access

- Users can access an array of information, stored across multiple sources, almost instantly. It means you will not be wasting time attempting to manually pull information from various sources, or seeking help from your IT department.

3. Consistent Quality Data

- Data warehouses gather information from countless sources and they convert it into a unified format to be used throughout your organisation. What does this mean? Well, you can have confidence that each of your departments will be producing results, which are in line, and consistent with each other, which in turn ensures company-wide accuracy.

https://docs.oracle.com/cd/E11882_01/server.112/e25555/tdpdw_intro.htm#TDPDW00100
https://dzone.com/articles/data-warehouse-characteristics-and-benefits
https://www.tutorialspoint.com/dwh/dwh_quick_guide.htm

3. Data Analytics

- Data analytics is a complex process of examining large and varied data sets, or big data, to uncover information -- such as **hidden patterns**, **unknown correlations**, **market trends** and **customer preferences** - that can help organisations make informed business decisions.
- Data analytics technologies and techniques provide a means to analyse data sets and draw conclusions about them which help organisations make informed business decisions.

Types of Data Analytics

1. **Descriptive analytics** - describes what has happened over a given period. Have the number of views gone up? Are sales stronger this month than last?
2. **Diagnostic analytics** - focuses more on why something happened. This involves more diverse data inputs and a bit of hypothesizing. Did the weather affect beer sales? Did that latest marketing campaign affect sales?
3. **Predictive analytics** - moves to what is likely going to happen in the near term. What happened to sales the last time we had a hot summer? How many weather models predict a hot summer this year?
4. **Prescriptive analytics** - suggests a course of action. If the likelihood of a hot summer is measured as an average of these five weather models is above 58%, we should add an evening shift to the brewery and rent an additional tank to increase output.

Data Analytic Tools

There are thousands of Big Data tools out there for data analysis today. They include

- **Data Visualization Tools**: They are used for making interactive charts. They include Datawrapper, Solver, Qlik, Infogram
- **Sentiment Tools:** Tools used to identify and evaluate subjective and expression sentiments within a textual content. This analysis may be performed at topical, sentence or document level. Examples of tools include Opentext, SAS Sentiment Analysis, Opinion Crawl
- **Data Extraction Tools**: They are used to extract information an individual needs from a website. Examples of these tools include Content Grabber, Import.io, Parsehub

- **Databases** : They hold information.

https://searchbusinessanalytics.techtarget.com/definition/big-data-analytics
https://www.mastersindatascience.org/resources/what-is-data-analytics/

Application of Big Data in the Society

Use by individuals;

- Health (e.g. fitness trackers, personalised healthcare, DNA testing),
- Transport (e.g.GPS, map apps, public transport apps, online travel booking),
- News (e.g. Twitter and Facebook and other social media),
- Entertainment (e.g. Netflix and Spotify and other streaming systems),
- Shopping (e.g. Amazon and online supermarkets and other online shopping systems),
- Chatbots/digital assistants (e.g. Siri, Cortana, Alexa, Google Assistant)

Use by organisations;

- Customer relations -Predictions of consumer behaviour
- Identifying risks and fraud prevention
- Optimising timings for (e.g. product launch/change, presenting offers/discounts),
- measuring use/success of digital products (e.g. online entertainment, computer games, websites),
- improving reliability/efficiency of engineering/scientific products (e.g. engines, mobile networks, traffic control systems)

Use by society;

- Economics (e.g. planning and modelling, taxation, money supply),
- Health (e.g. public health/medical systems, disease monitoring and control),
- Social media, communications (e.g. email, VoIP, mobile systems)

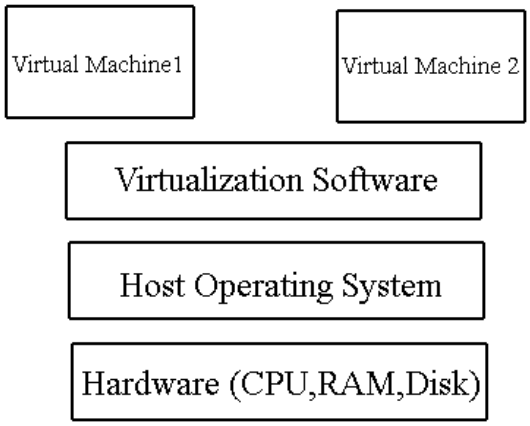
Resources
https://en.wikipedia.org/wiki/Examples_of_data_mining
https://hrcak.srce.hr/file/275281
https://www.simplilearn.com/tutorials/big-data-tutorial/big-data-applications
https://paulshepherd.co/innovative-ways-collect-big-data/
https://dataconomy.com/2014/06/understanding-big-data-infrastructure/
http://www.slate.com/articles/technology/technology/2017/10/what_happened_to_big_data.html?
https://www.edureka.co/blog/big-data-characteristics/
https://bigdata-madesimple.com/top-30-big-data-tools-data-analysis/

13. ENABLING TECHNOLOGIES

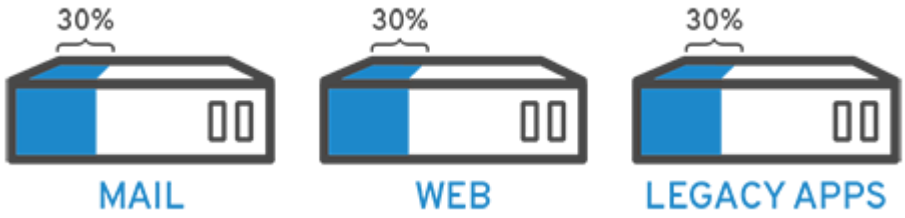
VIRTUALISATION

Have you ever think of running an operating system on top of another. If you are a Windows user, then you might want some great features of Linux OS. Similarly, if you are Linux user then you might want to work on MS Word. Virtualisation will cater all your needs be it software development, running one OS on another, configuration testing and document creation. It will allow you to install new Operating Systems on the same machine and the best thing is that you don't have to worry about hard disk formats and other pre install software or losing any data.

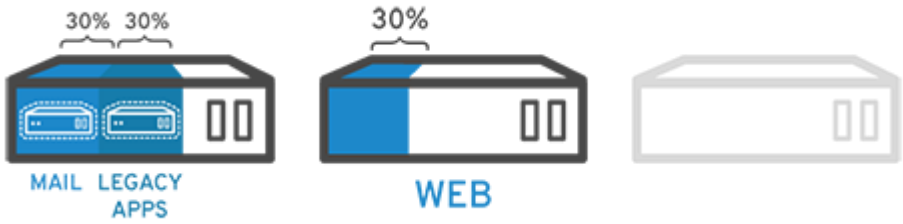
Virtualisation refers to running two or more operating systems on one physical PC. Either the multiple operating systems run side-by-side, with a separate piece of software called a **hypervisor** used to manage them, or one operating system runs the other operating systems within program windows. The former is usually limited to servers, with the latter finding common use on desktop computers.



In more practical terms, imagine you have 3 physical servers with individual dedicated purposes. One is a mail server, another is a web server, and the last one runs internal legacy applications. Each server is being used at about 30% capacity—just a fraction of their running potential. But since the legacy apps remain important to your internal operations, you have to keep them and the third server that hosts them.



Traditionally, it was often easier and more reliable to run individual tasks on individual servers (**1 server, 1 operating system, 1 task**.) It wasn't easy to give 1 server multiple brains. However, with Virtualisation, you can split the mail server into 2 unique ones that can handle independent tasks so the legacy apps can be migrated. It is the same hardware; you are just using more of it more efficiently.



Keeping security in mind, you could split the first server again so it could handle another task—increasing its use from 30%, to 60%, to 90%. Once you do that, the now empty servers could be reused for other tasks or retired altogether to reduce cooling and maintenance costs.

What can you do with Virtualisation?

1. Run Old Apps

Got an application that won't play nice in Windows 7 or Vista, but works fine in XP or an even earlier version of Windows, like Me? Just grab an old Windows CD and install it within a virtual machine (VM). Then install your app.

2. Access Virus-Infected Data

Ever been sent a file that your antivirus program has flagged, but which contains important data you just have to view? Most Virtualisation software includes snapshot functionality, which means you can create a "saved state" of the virtual OS and its entire hard disk. It's a little like travelling back in time.

You could create a snapshot in the virtual machine, open the infected file within the VM to access the data and, if the virus causes chaos, simply click to restore the VM snapshot. Hey presto--a clean virtualised computer.

3. Browse in Complete Safely

Why not install Windows on VMware Player, then install Firefox, and run it in Unity mode so it appears to run natively on the host computer?

Essentially Firefox will be running in what's known as a sandbox, meaning that should it (or one of its plugins) get hacked while you're online, there'll be no absolutely no risk to your actual operating system. You could create a snapshot once everything's been configured in the virtual machine in order to get things back up and running quickly, should anything go wrong.

4. Test Software, Upgrades, or New Configurations

The virus testing technique above isn't limited to malware. You could use your virtual computer to test new software, updates, or even new configurations of software before you roll them out for real on your main OS.

Some server administrators use Virtualisation to create a copy of an existing installation of an operating system, plus its data, which they then run virtualized and play with to see if configuration changes or updates will cause any harm. If you manage workstation computers and want to be sure a Windows update is OK before rolling it out, you could do the same thing--just test it in a virtualized machine first.

5. Run Linux on Top of Windows (or vice-versa)

Want to give Linux a try but can't face repartitioning your computer's hard disk? Provided that it would ordinarily install on your computer, you can run just about any operating system inside a virtual machine, including most Linux distros.

Linux and Mac users have been using Virtualisation for years to run Windows on top of their chosen OS in this way.
If you run a Linux machine for mail or Web services, as examples, having a desktop version of Linux for occasional use will also make it easier to communicate with the server. There's no need to install PuTTY on Windows to communicate via secure shell (SSH), for example, because Linux has that kind of thing built-in.

6. Back Up an Entire Operating System
Because the virtual OS is entirely contained within a series of files, backing it up is as simple as backing up any other files. It's the same with virtualised server installations too. If you're running a virtual machine on a server to host your mail server, and it's brought down by a hack attack, then bringing things back to working order is as simple as restoring the backup files (assuming the vulnerability that allowed the hack is quickly addressed once things are up and running, of course).
Bear in mind that creating a copy of a VM creates legal issues. Backing up should be fine, but if you create a copy of a VM installation to give to a friend, for example, then you'll be contravening copyright laws (assuming they apply, as with Microsoft, but not always with Linux).

7. Create a Personal Cloud Computer
If you're out of the office, there's no need to take your laptop with you. Just leave it running (with power saving turned off!), take your mobile phone or tablet computer instead, and access the laptop via a Remote Desktop Protocol (RDP) connection over the Internet. This will let you access the same desktop environment you're used to, although there'll be no fancy graphics.

8. Run Headless For Web Development

Most Virtualisation software allows the virtual machine to run headless, which is to say, without displaying a desktop (or other user-interface). Essentially, the virtual PC runs in the background although accepts all other kinds of connections, such as networking. For people creating websites, this offers the possibility of running their very own private web server for testing purposes.

9. Make a Backup of Your Server For Emergencies
Amazon's Elastic Cloud Computing (EC2) service allows you to copy across any existing virtualised Windows 2008 Server installation for use on EC2 (eventually all kinds of server installs will be supported, such as Linux).

Creating periodic backups of existing server installs in this way could provide vital redundancy if a catastrophe happens to your existing server. After the hurricane hits, leaving your physical servers in a whirl of dust, all you'd have to do is boot up the EC2 image, reconfigure things slightly to take into account the different IP addresses, and then continue as usual.

10. Reuse Old Hardware

By installing Citrix XenDesktop on your Windows server, you can turn old, less powerful computers into thin clients, wiping out the need for a workstation IT upgrade budget.

The clients access their personal desktop spaces on the server and there's little noticeable difference compared with running the operating system and applications locally. XenDesktop includes clever technology to avoid common thin-client pitfalls, such as the fact videos and animations don't play well, by shifting some of the processing work to the client computer.

XenDesktop also allows your workers to access their desktops from home, provided the server is configured to be publicly accessible and they have the right client software installed. You can even use mobile phones to connect to the desktop environments.

Advantages Of Virtualisation

- One of the biggest advantages of Virtualisation is scalability i.e. the ability to expand. Whenever there is excessive load on some part of application in a server you can easily create a similar virtual environment on a different server and configure the setup.
- Hardware maintenance cost is reduced because you don't need many servers to install different applications.
- You can save a huge amount of energy by running one physical server instead of many and less power backup is required.
- You can get faster and safer backups by taking live snapshot while server is running.
- You will get centralised monitoring of your resources as Virtualisation provides easy way of connecting and maintaining your virtual servers.

How Does Virtualisation Work?

Virtualisation is created with the help of software. This software is a program that creates an environment or platform wherein it emulates and imitates different hardware components like RAM, Hard disk, Network Card, Processor etc. Virtualisation creates an illusion where an Operating System starts believing that it is using or utilising the various hardware component of the computer system.
Virtualisation is achieved using software that creates a virtual environment which looks real but isn't. Following are some information you have to configure while creating a new Virtual Machine.

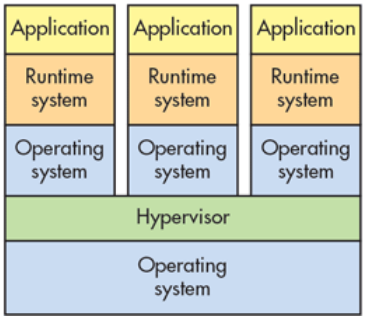
- You need to configure the amount of RAM which will be reserved by the software from the real RAM.
- You need to configure the size of hard disk you want to allocate to the virtual machine.
- You need to configure the number of network adapters the virtual machine will have.
- You need to tell the virtual machine whether it should access removable devices.
- You need to tell the virtual machine whether it can play audio or video.

When you are running an OS over the top of another on your machine the whole environment acts like a HOST and GUEST OS. The real operating system acts as a HOST and the OS run by virtualisation software acts as a GUEST OS.
The entire load balancing is actually done by the HOST operating system.

Ways to Achieve Virtualisation

- **Containerisation**
- **virtual machines**

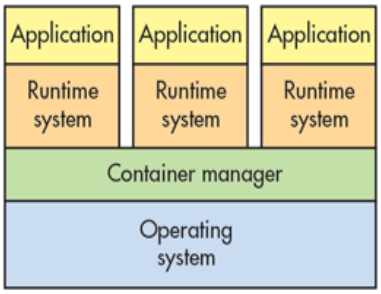
Virtual Machines



- VMs technique uses a special software called a **hypervisors** to allow virtualisation
- A hypervisor create a virtual instances of the OS, processor, storage, networking components
- This provides an abstract machine that uses device drivers targeting the abstract machine.
- A hypervisor runs VMs that have their own operating system using hardware VM support.
- The virtual machines emulate hardware, while containers emulate operating systems.

Containerisation

- A container is a virtual runtime environment that runs on top of a single operating system (OS) kernel and emulates an operating system rather than the underlying hardware. The technique creates a container for each application that is running
- The container's system requires an underlying operating system to provides the basic services to all of the containerised applications using virtual-memory support for isolation while a container provides an abstract OS
- Container systems have a lower overhead than VMs and container systems typically target environments where thousands of containers are in play.
- Container systems usually provide service isolation between containers. As a result, container services such as file systems or network support can have limited resource access.
- One advantage is the low overhead of containers and, therefore, the ability to start new containers quickly. This is because starting the underlying OS in a VM takes time, memory, and the space needed for the VM disk storage
- Containers are becoming more widely used because they provide many of the isolation benefits of VMs without as much time and space overhead.



https://www.youtube.com/watch?v=S7NVloq0EBc
https://www.docker.com/resources
https://opensource.com/resources/what-are-linux-containers
https://www.youtube.com/watch?v=FZR0rG3HKIk
https://www.youtube.com/watch?v=0qotVMX-J5s
https://www.youtube.com/watch?v=hPkEqOoQSu4
https://www.youtube.com/watch?v=7m3f-P-WWbg

Pros of Virtualisation via Containers

1. **Hardware costs**. Virtualisation via containers decreases hardware costs by enabling consolidation (i.e., the allocation of multiple applications to the same hardware improves hardware utilisation). It enables concurrent software to take advantage of the true concurrency provided by a multicore hardware architecture. In addition, it enables system architects to replace several lightly-loaded machines with fewer, more heavily-loaded machines to minimise SWAP-C (size, weight, power, and cooling), free up hardware for new functionality, support load balancing, and support cloud computing, server farms, and mobile computing.
2. **Reliability and robustness**. The modularity and isolation provided by VMs improve reliability, robustness, and operational availability by localising the impact of defects to a single VM and enabling software failover and recovery.
3. **Scalability**. A single container engine can efficiently manage large numbers of containers, enabling additional containers to be created as needed.
4. **Spatial isolation**. Containers support lightweight spatial isolation by providing each container with its own resources (e.g., core processing unit, memory, and network access) and container-specific namespaces.
5. **Storage**. Compared with virtual machines, containers are lightweight with regard to storage size. The applications within containers share both binaries and libraries.
6. **Performance**. Compared with virtual machines, containers increase performance (throughput) because they do not emulate the underlying hardware.
7. **Real-time applications**. Containers provide more consistent timing than virtual machines
8. **Continuous integration**. Containers support Agile and continuous development processes by enabling the integration of increments of container-hosted functionality.
9. **Portability**. Containers support portability from development to production environments, especially for cloud-based applications.
10. **Safety**. Safety is improved by localising the impact of faults and failures to individual containers.
11. **Security**. The modular architecture provided by the containers increases the complexity and difficulty of attacks. Spatial isolation largely limits impact of malware to a single container. A container that is compromised can be terminated and replaced with a new container that is booted from a known clean image, which enables a rapid system restore or software reload following a cybersecurity compromise. Finally, security software and rules implemented at the container engine level can apply to all of its containers.

Cons of Virtualisation via Containers

1. **Shared Resources**. Applications within containers share many resources including (1) container-specific resources including the container, container engine, and OS kernel, (2) Virtualisation by VM resources including the virtual machine, the hypervisor, and the host operating system if using a type-2 hypervisor, and (3) multicore processing resources including (a) *processor-internal* resources (L3 cache, system bus, memory controller, I/O controllers, and interconnects) and (b) *processor-external* resources (main memory, I/O devices, and networks). Such shared resources imply (1) single points of failure exist, (2) two applications running in the *same container* can interfere with each

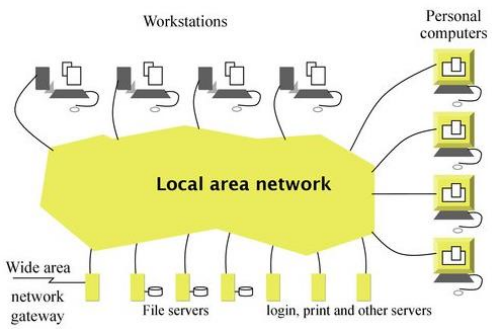
other, and (3) software running in *one container* can impact software running in *another container* (i.e., interference can violate spatial and temporal isolation).

2. **Interference Analysis**. Shared resources imply interference. Virtualisation via containers increases the difficulty of the analysis of temporal interference (e.g., meeting timing deadlines) and tends to make the resulting timing estimates overly conservative. Interference analysis becomes more complex as the number of containers increases and Virtualisation via containers is combined with Virtualisation via VMs and multicore processing. The number of interference paths increases rapidly as the number of containers increases. The resulting large number of interference paths typically make the exhaustive analysis of all such paths impossible. This, in turn, forces the architect to select only a relatively small representative selection of paths.
3. **Safety**. Although containers provide significant support for isolation, the use of containers does not guarantee isolation. Temporal interference may cause delays that cause hard, real-time deadlines to be missed. Spatial interference can cause memory clashes. Whether by VM or containers, traditional safety accreditation and certification processes and policies do not take Virtualisation into account, making safety (re)accreditation and (re)certification difficult.
4. **Security**. Containers are not by default secure and require significant work to make them secure. One must to ensure that (1) no data is stored inside containers, (2) container processes are forced to write to container-specific file systems, (3) the container's network namespace is connected to a specific, private intranet, and (4) the privileges of container services are minimised (e.g., non-root if possible). As with safety, traditional security accreditation and certification processes and policies do not take Virtualisation into account, making security (re)accreditation and (re)certification difficult.
5. **Container Sprawl**. Excessive containerisation is relatively common and increases the amount of time and effort spent on container management.

DISTRIBUTED SYSTEM

- The pace at which computer systems change was, is, and continues to be overwhelming.
- From 1945, when the modern computer era began, until about 1985, computers were large and expensive. Moreover, for lack of a way to connect them, these computers operated independently from one another.
- Starting in the mid-1980s, however, two advances in technology began to change that situation. The first was the development of **powerful microprocessors**. Initially, these were 8-bit machines, but soon 16-, 32-, and 64-bit CPUs became common.
- With **multicore CPUs**, we now are re-facing the challenge of adapting and developing programs to exploit parallelism. In any case, the current generation of machines have the computing power of the mainframes deployed 30 or 40 years ago, but for 1/1000th of the price or less.
- The second development was the invention of high-speed computer networks. **Local-area networks** or **LANs** allow thousands of machines within a building or campus to be connected in such a way that small amounts of information can be transferred in a few microseconds or so. Larger amounts of data can be moved between machines at rates of billions of *bits per second* (bps). **Wide-area networks** or **WANs** allow hundreds of millions of machines all over the earth to be connected at speeds varying from tens of thousands to hundreds of millions bps, and sometimes even faster.
- Parallel to the development of increasingly powerful and networked machines, we have also been able to witness **miniaturisation** of computer systems with perhaps the smartphone as the most impressive outcome. Packed with sensors, lots of memory, and a powerful CPU, these devices are nothing less than full-fledged computers. Of course, they also have networking capabilities. Along the same lines, **plug computers** and other so-called **nano computers** are finding their way to the market. These small computers, often the size of a power adapter, can often be plugged directly into an outlet and offer near-desktop performance.
- The result of these technologies is that it is now not only feasible, but easy, to put together a computing system composed of many networked computers, be they large or small. These computers are generally geographically dispersed, for which reason they are usually said to form a **distributed system**. The size of a distributed system may vary from a handful of devices, to millions of computers. The interconnection network may be wired, wireless, or a combination of both. Moreover, distributed systems are often highly dynamic, in the sense that computers can join and leave, with the topology and performance of the underlying network almost continuously changing.

A Distributed System



Definition 1

- A distributed system can be defined as a collection of autonomous computing elements that appears to its users as a single coherent system.
- This definition refers to two characteristic features of distributed systems.
    - The first one is that a distributed system is **a collection of computing elements each being able to behave independently of each other**. A computing element (**node)** can be either a hardware device or a software process.
    - A second element is that users (be they people or applications) believe they are dealing with a single system. This means that one way or another the autonomous nodes need to collaborate.

Characteristic 1: Collection of Autonomous Computing Elements

- Modern distributed systems will consist of all kinds of nodes, ranging from very big high-performance computers to small plug computers or even smaller devices. A fundamental principle is that nodes can act independently from each other, although it should be obvious that if they ignore each other, then there is no use in putting them into the same distributed system. In practice, nodes are programmed to achieve common goals, which are realised by exchanging messages with each other. A node reacts to incoming messages, which are then processed and, in turn, leading to further communication through message passing.
- An important observation is that, as a consequence of dealing with independent nodes, each one will have its own notion of time. In other words, we cannot assume that there is something like a **global clock**. This lack of a common reference of time leads to fundamental questions regarding the synchronisation and coordination within a distributed system.

2.2 Characteristic 2: single coherent system

- As mentioned, a distributed system should appear as a single coherent system. In some cases, researchers have even gone so far as to say that there should be a single-system view, meaning that an end user should not even notice that processes, data, and control are dispersed across a computer network. Achieving a single-system view is often asking too much, for which reason, in our definition of a distributed system, we have opted for something weaker, namely that it *appears* to be coherent. Roughly speaking, a distributed system is coherent if it behaves according to the expectations of its users. More specifically, in a single coherent system the collection of nodes as a whole operates the same, no matter where, when, and how interaction between a user and the system takes place.
- Offering a single coherent view is often challenging enough. For example, it requires that an end user would not be able to tell exactly on which computer a process is currently executing, or even perhaps that part of a task has been spawned off to another process executing somewhere else. Likewise, where data is stored should be of no concern, and neither should it matter that the system may be replicating data to enhance performance. This so-called **distribution transparency** is an important design goal of distributed systems. In a sense, it is akin to the approach taken in many Unix-like operating systems in which resources are accessed through a unifying file-system interface, effectively hiding the differences between files, storage devices, and main memory, but also networks.

Definition 2

- A distributed system is a group of computers working together as to appear as a single computer to the end-user.
- These machines have a **shared state**, **operate concurrently** and can **fail independently** without **affecting the whole system's uptime**.

Examples:

Distributed systems

- Department computing cluster
- Corporate systems
- Cloud systems (e.g. Google, Microsoft, etc.)

Application examples

- Email
- News
- Multimedia information systems - video conferencing
- Airline reservation system
- Banking system
- File downloads (BitTorrent)
- Messaging

Why Distributed System/Advantages?

- **Scaling Horizontally**
  - For a machine to handle more traffic we would upgrade the hardware i.e. processor, hard disk etc. But all these upgrades have a limit. This is **scaling vertically**.
  - **Scaling horizontally** simply means adding more computers rather than upgrading the hardware of a single one.
  - It is significantly cheaper than vertical scaling after a certain threshold
  - The best thing about horizontal scaling is that you have no cap on how much you can scale — whenever performance degrades you simply add another machine, up to infinity potentially.
- **Fault Tolerance**
  - In Fault tolerance, in case a single machine stops to function, other systems pick up the function of the other machine and the system works as if there was problem
  - A cluster of ten machines across two data centres is inherently more fault-tolerant than a single machine. Even if one data centre catches on fire, your application would still work.
- **Low Latency**
  - The time for a network packet to travel the world is physically bounded by the speed of light. For example, the shortest possible time for a request‘s round-trip time (that is, go back and forth) in a fiber-optic cable between New York to Sydney is 160ms. Distributed systems allow you to have a node in both cities, allowing traffic to hit the node that is closest to it.
- **Cost Savings**
  - There is no need of purchasing the state of the art machines to include within the system. Because a task is split between the different systems, any computer can join the system. So even old machines that may not be able to handle large software can still be used to carry out processing.
- **Reliability And Resilience**
  - Failure of a single node or computer within the system does not affect other systems. Other system continue to carry on their activities as if nothing happens. So a process proceeds on without any hitch.
- **Availability.**
  - Because the system consists of several machines, the system is always running at any time. Failure of a single machine does not stop the whole system. At any time one can use the system.

- **Speed and Content Distribution**
  - Queries can be routed to a server with a particular user's information, rather than all requests having to go to a single machine that can be overloaded.
  - Requests can also be routed to servers physically close or on a speedy network connection to whoever wants the data, which can mean less time and other resources allocated to dealing with network traffic and bottlenecks.

- http://csis.pace.edu/~marchese/CS865/Lectures/Chap1/Chapter1a.htm
- https://www.freecodecamp.org/news/a-thorough-introduction-to-distributed-systems-3b91562c9b3c/?gi=9c6ca5d69e2
- https://www.youtube.com/watch?v=Y6Ev8GIlbxc
- http://www.cs.fsu.edu/~xyuan/cop5611/lecture2.html
- https://ds.cs.luc.edu/issues/issues.html

Categories of Distributed Systems;

1. **Distributed databases**

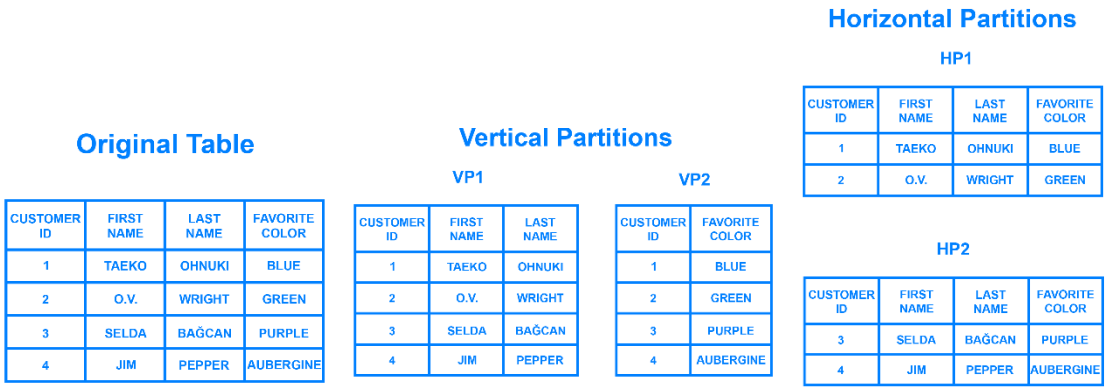
A distributed database is a database that consists of two or more files located in different sites either on the same network or on entirely different networks. Portions of the database are stored in multiple physical locations and processing is distributed among multiple database nodes. This means that similar data can be accessed at the same time, by different peoples and at different locations. The system must ensure that a user's action/activity does not affect another user's activity. This is termed as **ACID**.

ACID means **Atomicity, Consistency, Isolation** and **Durability**

- **Atomicity** means that you guarantee that either all of the transaction succeeds or none of it does. You don't get part of it succeeding and part of it not. If one part of the transaction fails, the whole transaction fails. With atomicity, it's either "all or nothing". For example if a user access a record, the user must have exclusive rights to that record and update the record. If the user can only update part of the record the system should deny the user the right to do that
- **Consistency** ensures that you guarantee that all data will be consistent. All data will be valid according to all defined rules, including any constraints, cascades, and triggers that have been applied on the database.
- **Isolation** guarantees that all transactions will occur in isolation. No transaction will be affected by any other transaction. So a transaction cannot read data from any other transaction that has not yet completed.
- **Durability** means that, once a transaction is committed, it will remain in the system – even if there's a system crash immediately following the transaction. Any changes from the transaction must be stored permanently. If the system tells the user that the transaction has succeeded, the transaction must have, in fact, succeeded.

2. **Sharding / Partitioning Database**

This is a database architecture pattern related to separating one table's rows/columns into multiple different tables, known as partitions. Each partition has the same design and columns/rows, but also entirely different rows/columns. Likewise, the data held in each is unique and independent of the data held in other partitions. Database may be divided vertically where (**vertical partition**) or horizontally (**Horizontal partition**)



https://www.digitalocean.com/community/tutorials/understanding-database-sharding

3. **Distributed File Systems** (e.g. nodes, blockchain)

This is a method of storing and accessing files based in a **client/server architecture**. In a distributed file system, one or more central servers store files that can be accessed by any number of remote clients in the network.

Much like an operating system organises files in a hierarchical file management system, the distributed system uses a uniform naming convention and a mapping scheme to keep track of where files are located. When the client device retrieves a file from the server, the file appears as a normal file on the client machine, and the user is able to work with the file in the same ways as if it were stored locally on the workstation. When the user finishes working with the file, it is returned over the network to the server, which stores the now-altered file for retrieval at a later time.

Distributed file systems can be advantageous because they make it easier to distribute documents to multiple clients and they provide a centralized storage system so that client machines are not using their resources to store files.

4. **Distributed Applications** (e.g. Online Games, Web Browsers, Search Engines)

Distributed applications (distributed apps) are **applications** or **software** that runs on multiple computers within a network at the same time and can be stored on servers or with cloud computing. Unlike traditional applications that run on a single system, distributed applications run on multiple systems simultaneously an effort to complete a specific task or command.

For example in the **cryptoeconomy**, the **blockchain** used by most cryptocurrencies uses Distributed Apps to maintain an efficient digital marketplace. Rather than the conventional client-server network adopted by most centralized organizations, blockchains run on a peer-to-peer network where transactional information carried out between two parties is recorded and shared across multiple computers on the network. These computers are referred to as nodes. Each node acts as an administrator in the Bitcoin markets and joins the network voluntarily for the opportunity to receive Bitcoins as a reward.

Each node has a duplicate copy of an original transaction, which gets continually reconciled by the network. So whatever entry that node A has on its record for a Bitcoin transaction between Jane and John cannot differ from what nodes B, C, D, E, and F have. This means of verifying each transaction by multiple nodes is called **distributed ledgers**. This means that since a version of events can be verifiable with different computers, a hacker, even though he gets into one system to tweak the transaction, would need to get into all the systems spread across various geographical locations to corrupt the recorded data.

Designing a Distributed System / Issues with Distributed System

There are some important issues that should be considered when creating a distributed system. They are

1. Supporting resource sharing

- An important goal of a distributed system is to make it easy for users (and applications) to access and share remote resources.
- Resources include peripherals, storage facilities, data, files, services, and networks, to name just a few. One reason for sharing resources is that of economics. For example, it is cheaper to have a single high-end reliable storage facility be shared than having to buy and maintain storage for each user separately.

2. Making Distribution Transparent/Invisible to the End users

- An important goal of a distributed system is to hide the fact that its processes and resources are physically distributed across multiple computers, possibly separated by large distances. In other words, it tries to make the distribution of processes and resources **transparent**, that is, invisible, to end users and applications.
- There are different types of transparency. They include

Transparency	Description
Access	Hide differences in data representation and how an object is accessed
Location	Hide where an object is located
Relocation	Hide that an object may be moved to another location while in use
Migration	Hide that an object may move to another location
Replication	Hide that an object is replicated
Concurrency	Hide that an object may be shared by several independent users
Failure	Hide the failure and recovery of an object

- Although distribution transparency is generally considered preferable for any distributed system, there are situations in which attempting to blindly hide all distribution aspects from users is not a good idea. A simple example is requesting your electronic newspaper to appear in your mailbox before 7 A.M. local time, as usual, while you are currently at the other end of the world living in a different time zone. Your morning paper will not be the morning paper you are used to.

3. Being Open

- Another important goal of distributed systems is openness. An **open distributed system** is essentially a system that offers components that can easily be used by, or integrated into other systems. At the same time, an open distributed system itself will often consist of components that originate from elsewhere.

4. Failure

- Computer systems sometimes fail. When faults occur in hardware or software, programs may produce incorrect results or may stop before they have completed the intended computation. The handling of failures is particularly difficult when dealing with distributed system. If one node fails how do you transfer the task to the other computer?

5. Concurrency

- Both services and applications provide resources that can be shared by clients in a distributed system. There is therefore a possibility that several clients will attempt to access a shared resource at the same time.
- For example, in a school, the marks files may be accessed and be edited by several teachers during the end of the term. The system must synchronise these activities in such a way that the data remain consistent.

6. Security

- Many of the information resources that are made available and maintained in distributed systems have a high intrinsic value to their users. Their security is therefore of considerable importance. Security for information resources has three components:
  confidentiality (protection against disclosure to unauthorized individuals)
  integrity (protection against alteration or corruption),
  availability for the authorized (protection against interference with the means to access the resources)

https://www.ejbtutorial.com/distributed-systems/challenges-for-a-distributed-system

Limitations of Distributed Systems

1. Absence of a Global Clock

In a distributed system there are a lot of systems and each system has its own clock. Each clock on each system is running at a different rate leading to them asynchronous (at different rate).
At the beginning, the clocks are regulated to keep them consistent, but only after one local clock cycle they are out of the synchronization and no clock has the exact time.
Time is known for a certain precision because it is used for the following in distributed system:

- Temporal ordering of events
- Collecting up-to-date information on the state of the integrated system
- Scheduling of processes

There are restrictions on the precision of time by which processes in a distributed system can synchronize their clocks due to asynchronous message passing. Every clock in distributed system is synchronize with a more reliable clock, but due to transmission and execution time lapses the clocks becomes different. Absence of global clock make more difficult the algorithm for designing and debugging of distributed system.

2. Absence of Shared Memory:

Distributed systems have not any physically shared memory, all computers in the distributed system have their own specific physical memory. As computer in the distributed system do not share the common memory, it is impossible for any one system to know the global state of the full distributed system. Process in the distributed system obtains coherent view of the system but in actual that view is partial view of the system.
As in distributed system there is an absence of a global state, it is challenging to recognize any global property of the system. The global state in distributed system is divided by many number of computers into smaller entities.

3. Security

As the numbers of nodes increases so does the opportunities of attacks increase also. Each node implements its own security measures which may or may not be adequate. There is no stand security measures for all the nodes.
Also as the nodes are added.so does the opportunity for inside attack from the weakest link. This may provide access to an entire system

4. Performance;

There are several machines within a distributed system. These machines are linked together via a network architecture. A distributed algorithm and protocols needs to be developed to manage these machines and processes.
Because data is being transferred between different machines, there is need for a network that has a bigger bandwidth so as to ensure updating and data transfer takes place at a faster rate or a low latency levels. This is a big issue when dealing with distributed systems

Resources

An introduction to distributed systems.
https://medium.freecodecamp.org/a-thorough-introduction-to-distributed-systems-3b91562c9b3c?gi=9c6ca5d69e2

Hadoop open-source software for distributed computing, download and documentation.
https://hadoop.apache.org/#

13.3.1 HUMAN COMPUTER INTERACTION

- Until the late 1970s, the only humans who interacted with computers were information **technology professionals** and **dedicated hobbyists**. This changed disruptively with the emergence of personal computing in the later 1970s.
- Personal computing, including both personal software and personal computer platforms made everyone in the world a potential computer user
- The original focus of HCI was and is the concept of *usability*. This meant making the computer system **"easy to learn, easy to use".**
- However, the concept of usability has been re-articulated and reconstructed almost continually, and has become increasingly rich and intriguingly problematic. Usability now often subsumes qualities like **fun**, **well-being**, **collective efficacy**, **aesthetic tension**, **enhanced creativity, flow, support for human development**, and others.
- Although the original academic home for HCI was computer science, and its original focus was on personal productivity applications, mainly text editing and spreadsheets, the field has constantly diversified and outgrown all boundaries. It quickly expanded to encompass visualisation, information systems, collaborative systems, the system development process, and many areas of design.
- HCI expanded from desktop office applications to include games, learning and education, commerce, health and medical applications, emergency planning and response, and systems to support collaboration and community.
- It has expanded from early graphical user interfaces to include myriad interaction techniques and devices, multi-modal interactions, tool support for model-based user interface specification, and a host of emerging ubiquitous, handheld and context-aware interactions.
- HCI academic programs train many different types of professionals: user experience designers, interaction designers, user interface designers, application designers, usability engineers, user interface developers, application developers and technical communicators/online information designers

Beyond the desktop

- One of biggest design ideas of the early 1980s was use of icons to represent **Files** and **folders**. This lead to soon **double clicking**, **dragging** windows and icons around the displays
- As HCI developed, it moved beyond the desktop in three distinct senses.

    - First, representing digital objects as **icons**. This lead to cluttering of the desktop.
    - The second, the **growing influence of the Internet on computing and on society**. Emails emerged as one of the most important HCI applications. People ceased to interact with computers but other people through computers. Tools and applications to support collaborative activity now include instant messaging, wikis, blogs, online forums, social networking, social bookmarking and tagging services, media spaces and other collaborative workspaces. This area of HCI, now often called **social computing**.
    - The third way that HCI moved beyond the desktop was through **the continual and occasionally explosive diversification in the ecology of computing devices**. Before desktop applications were consolidated, new kinds of device contexts emerged, notably laptops, which began to appear in the early 1980s, and handhelds, which began to appear in the mid-1980s. One frontier today is ubiquitous computing: The pervasive incorporation of computing into human habitats — cars, home appliances, furniture, clothing, and so forth.
- The focus of HCI has moved beyond the desktop, and its focus will continue to move.
- Today, we are seeing departures from the **keyboard**-and-**mouse** interface configuration. Touchscreen devices like smartphones and tablet computers have introduced this technology -- which has been around for more than a decade -- to a wider audience.
- We are also making smaller computers, which necessitates new approaches to user interfaces. You would not want a full-sized keyboard attached to your smartphone -- it would ruin the experience.
- Touchscreens have introduced new techniques for computer navigation. Early touchscreens could only detect a single point of contact -- if you tried touching a display with more than one finger, it could not follow your movements. But today, you can find multi-touch screens in dozens of computer devices. Engineers have taken advantage of this technology to develop gesture navigation.
- Users can execute specific commands with predetermined gestures. For example, several touchscreen devices like the Apple iPhone allow you to zoom in on a photo by placing two fingers on the screen and drawing them apart. Pinching your fingers together will zoom out on the photo.
- Today, devices like smartphones can transcribe voice messages into text messages with variable accuracy. And there are already applications that allow users to control devices through vocal commands.
- Other engineers are working on an entirely different hands-free interface. It does not require the user touching a computer screen. The system accomplishes this using a collection of sensors and cameras to interpret a user's movements and translate them into computer commands.
- The user wears a special pair of gloves that have reflective beads on them. The cameras track the movement of the beads to interpret the user's movements.
- Your interactions with computer systems may even become passive. Using radiofrequency identification (RFID) tags, you can interact with computer systems just by being near them. This technology can have harmless, fun uses like tracking you as you walk through an environment so that your favourite kind of music plays in each room or the climate control system adjusts to your preselected preferences. Or it could be used for surveillance purposes to track people as they move through an environment.
- It can also help you cook dinner. Imagine bringing home a collection of ingredients, each of which has an RFID tag. Your home's integrated computer system detects what you have brought and determines you want to make prepare. Instantly, your home produces the recipe and asks if you want to preheat your oven
- You may not need RFID chips at all. Microsoft's Kinect peripheral for the Xbox 360 uses cameras to map out the environment in front of the entertainment centre. As a user steps in front of the camera, the system maps the user's frame and face and allows the user to create a

profile. Then, whenever that person steps into frame, the system knows who it is. The profile can store user preferences and skill levels so you do not have to worry about jumping into a game and getting your character slaughtered five seconds into it.

https://www.getsmarter.com/blog/market-trends/14-human-computer-interaction-examples/
https://www.interaction-design.org/literature/topics/human-computer-interaction
https://www.tutorialspoint.com/human_computer_interface/quick_guide.htm

Ways Users Interact with the Computers

Visual interaction

- This refers to the adoption of user interfaces for interactive systems that make use of visual elements and visual interaction strategies with the aim of supporting perceptual inferences instead of arduous cognitive (mind) comparisons and computations.
- The design of Visual Interaction focuses on the definition of interaction mechanisms through which
    - the users can perform actions on the interactive system by means of visual elements
    - the system can provide feedback to the user, by visually representing the results of the computations triggered by the user actions.
- Visual interaction is achieved through the use of **flat screens, 3d/virtual reality screens, eye tracking, head/hand/body tracking, virtual retinal display/retinal projector, head-up displays**

https://www.interaction-design.org/literature/topics/visual-design
https://www.tutorialspoint.com/human_computer_interface/quick_guide.htm
https://psu.pb.unizin.org/ist110/chapter/5-2-human-computer-interaction/

Audio interaction

- Refers to the use of sound to pass messages between the user and the digital device
- Requires microphone and sometimes a speaker
- Resources that makes it possible to use audio interaction include microphones, speakers, voice commands, natural language, language translation, audio implants

Haptic interaction

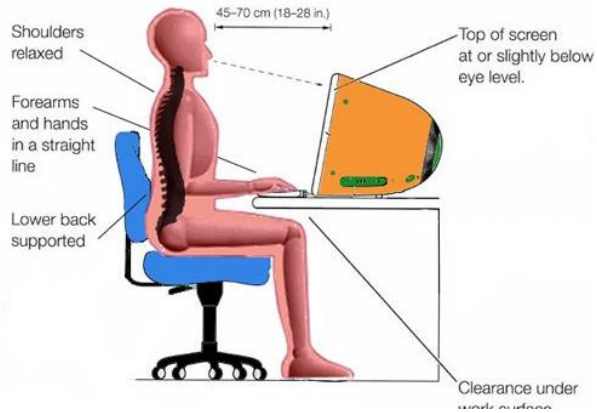
- A **haptics interface** is a system that allows a human to interact with a computer through bodily sensations and movements.
- Haptic encompasses tactile feedback or other bodily sensations to perform actions or processes on a computing device.
- A haptic device is one that involves physical contact between the computer and the user, usually through an input/output device, such as a joystick or data gloves, that senses the body's movements.
- By using haptic devices, the user can not only feed information to the computer but can receive information from the computer in the form of a felt sensation on some part of the body.
- For example, in a virtual reality environment, a user can pick up a virtual tennis ball using a data glove. The computer senses the movement and moves the virtual ball on the display. However, because of the nature of a haptic interface, the user will feel the tennis ball in his hand through tactile sensations that the computer sends through the data glove, mimicking the feel of the tennis ball in the user's hand.
- Devices used in Haptic Interaction include **joystick, mouse, trackball, keyboard, touch screen, haptic gloves and other wearables, force feedback devices, Braille displays**

Insertions

- Requires a chip being inserted into body of the individual
- The chip contains the user information
- When the chip is detected the system recalls data related to that individual and the mood the individual is in.
- It uses the RFID and emerging technology such as machine learning and natural language processing.

https://www.tutorialspoint.com/human_computer_interface/quick_guide.htm

Ergonomics


45–70 cm (18–28 in.)
Shoulders relaxed
Top of screen at or slightly below eye level.
Forearms and hands in a straight line
Lower back supported
Clearance under

- Many people spend hours a day in front of a computer without thinking about the impact on their bodies.
- They physically stress their bodies daily without realising it by extending their wrists; slouching, sitting without foot support and straining to look at poorly placed monitors.
- These practices can lead to cumulative trauma disorders or repetitive stress injuries, which create a life-long impact on health. Symptoms may include pain, muscle fatigue, loss of sensation, tingling and reduced performance.
- Ergonomics is a field of study that attempts to reduce strain, fatigue, and injuries by improving product design and workspace arrangement. The goal is a comfortable, relaxed posture.
- The term is derived from the Greek words '**Ergon'** meaning work and '**nomos'** meaning laws.
- Thus, ergonomics can be simply defined as the how workplace and equipment can be best used and designed for comfort, safety, efficiency and productivity.
- *Simply put, Ergonomics is the branch of science that deals with the people and their working environment.*

Importance of Ergonomics:

a) Increases productivity

- Best ergonomic solution enhances the productivity by reducing the unwanted tension, awkward position of the body. It is focused in making the work environment comfortable, this thereby reduces any kind of stress, risk and enhances the satisfaction and productivity.

b) Reduces the cost

- As ergonomics is focused about marinating the better health of the worker it reduces the cost of compensation that would be made by the injured or unhealthy staffs.
- It also reduces the indirect and the opportunity cost that could have incurred due to injury.

c) Improves the quality of the work

- Improved ergonomics favours the favourable environment where the workers can work efficiently.
- As the ergonomics improves level of satisfaction in the quality of the work increases.

d) Others

- Helps to reduce the absenteeism due to more comfort, safety and healthy working environment
- Assurance to the worker as their workplace is safer (acts as the motivation)
- More focus on the working environment and worker's health makes them feel valued and boost of moral.

Principles of Ergonomics

There are 10 fundamental principles of ergonomics, which are:

1. Work in neutral postures

- Proper posture maintenance is necessary
- Working too long with "C" curve can cause strain
- Keeping the proper alignment of neck hands wrist are also necessary

2. Reduce excessive force

- Excessive pressure or force at the joints can cause injury
- Better to minimize the work that requires more physical labour

3. Keep everything in reach

- Keeping everything in reach would help in avoiding unneeded stretching and strain
- More or less this principle is related with maintaining good posture.

4. Work at proper height

- Working at right makes things way easier
- Sometimes height can be maintained by adding extensions or avoiding extensions on the chair or tables

5. Reduce excessive motions

- Repetitive motion needs to be avoided
- This can cause disorder and numbness in long run
- Motion scan be reduced by the use of power tools

6. Minimize fatigue and static load

- Fatigue is common in strenuous work
- Having to hold things for longer period is example of static load
- Fatigue can be reduced by the intervals and the breaks between the works.

7. Minimize pressure points

- One needs to be aware of pressure points
- Almost everyone of has to sit on chairs that had cushioning, one of the pressure point is behind knees, which happens if air is too high or when you dangle your legs. Pressure point is also created in between your thigh and the bottom of a table when you sit.
- Anti-fatigue mats or insole can be used

8. Provide clearance

- Work area should have enough clearance
- Let the worker not worry about the bumps that they have to encounter on daily basis.

9. Move, exercise and stretch

- Move and stretch when you can
- It better to take intervals between the works and stretch and move along
- Stretching technique may differ and depend on the work one does

10. Maintain a comfortable environment

- This principle is focused on the other component of the working environment.
- It is concerned about the lightening, space, cool air and many more.

User Interface Design Basics

User Interface Design focuses on anticipating what users might need to do and ensures that the interface are easy to access, understand, and use.

Because users are familiar with user interfaces acting in a specific manner, it is important to maintain consistency and predictability of how and interface will react under the user's direction.

When creating a user interface consider the following

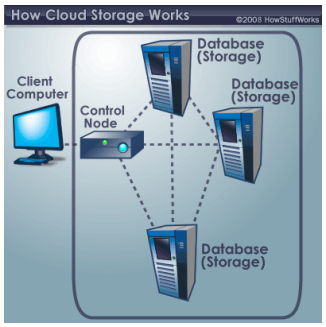
1. **Keep the interface simple**. Make the user interface to be as simple as possible so that the user does not have to spend a lot of time learning the interface.
2. **Be consistent**. By using common elements, users feel more comfortable and are able to get things done more quickly. Create patterns in language, layout and design throughout the site to help facilitate efficiency. Once a user learns how to do something, they should be able to transfer that skill to other parts of the site.
3. **Be purposeful in page layout**. Consider the spatial relationships between items on the page and structure the page based on importance. Careful placement of items can help draw attention to the most important pieces of information and can aid scanning and readability.
4. **Strategically use colour and texture**. You can direct attention toward or redirect attention away from items using colour, light, contrast, and texture to your advantage.
5. **Use typography to create hierarchy and clarity.** Carefully consider how you use typeface. Different sizes, fonts, and arrangement of the text to help increase scanability, legibility and readability.
6. **Make sure that the system communicates what's happening**. Always inform your users of location, actions, changes in state, or errors. The use of various UI elements to communicate status and, if necessary, next steps can reduce frustration for your user.

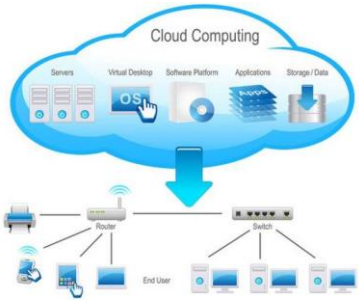
How are the Ergonomic Principles used in the Interface design of windows operating system items such as

- **menus**
- **icons**
- **accessibility**
- **windows**
- **pointers**

13.4 STORING AND SECURING DATA

13.4.1 CLOUD STORAGE





Definition:

- This is saving data to an off-site storage system maintained by a third party. Instead of storing information to your computer's hard drive or other local storage device, you save it to a remote database.
- This is a service model in which data is transmitted and stored on remote storage systems, where it is maintained, managed, backed up and made available to users over a network
- The Internet provides the connection between your computer and the database.

Cloud Storage Basics

- The facilities that house cloud storage systems are **data centres**.
- At its most basic level, a cloud storage system needs just one data server connected to the Internet.
- A client (e.g., a computer user subscribing to a cloud storage service) sends copies of files over the Internet to the data server, which then records the information.
- When the client wishes to retrieve the information, he or she accesses the data server through a Web-based interface. The server then either sends the files back to the client or allows the client to access and manipulate the files on the server itself.
- Cloud storage systems generally rely on hundreds of data servers. Because computers occasionally require maintenance or repair, it is important to store the same information on multiple machines. This is called **redundancy**. Without redundancy, a cloud storage system could not ensure clients that they could access their information at any given time.
- **Cloud storage is a subcategory of cloud computing. Cloud computing systems offer users access to not only storage, but also processing power and computer applications installed on a remote network**.

Examples of Cloud Storage

- **Google Docs** allows users to upload documents, spreadsheets and presentations to Google's data servers. Users can edit files using a Google application. Users can also publish documents so that other people can read them or even make edits, which means Google Docs is also an example of cloud computing.
- **Web e-mail providers** like **Gmail**, **Hotmail** and **Yahoo**! Mail store e-mail messages on their own servers. Users can access their e-mail from computers and other devices connected to the Internet.
- Sites like **Flickr** and **Picasa** host millions of digital photographs. Their users create online photo albums by uploading pictures directly to the services' servers.
- **YouTube** hosts millions of user-uploaded video files.

- Web site hosting companies like **StartLogic**, **Hostmonster** and **GoDaddy** store the files and data for client Web sites.
- **Social networking sites** like **Facebook** and **MySpace** allow members to post pictures and other content. All of that content is stored on the respective site's servers.
- Services like **Xdrive**, **MediaMax** and **Strongspace** offer storage space for any kind of digital data.

Types of cloud storage

There are three main cloud-based storage access models: public, private and hybrid.

Public Cloud Storage services provide a Multi-tenant storage environment that is most suited for unstructured data on a subscription basis. Data is stored in the service providers' data centers with storage data spread across multiple regions or continents.
Customers generally pay on a per-use basis similar to the utility payment model
This market sector is dominated by **Amazon Simple Storage Service** (S3), **Amazon Glacier for cold or deep archival storage, Google Cloud Storage, Google Cloud Storage Nearline for cold data** and **Microsoft Azure.**

Private Cloud Storage service is provided by in-house storage resources deployed as a dedicated environment protected behind an organization's firewall.
Private clouds are appropriate for users who need customization and more control over their data, or who have stringent data security or regulatory requirements.

Hybrid Cloud is a mix of private cloud storage and third-party public cloud storage services. The model offers businesses flexibility and more data deployment options. An organization might, for example, store actively used and structured data in an on-premises cloud, and unstructured and archival data in a public cloud. A hybrid environment also makes it easier to handle seasonal or unanticipated spikes in data creation to the external storage service and avoiding having to add in-house storage resources.

https://www.cleo.com/blog/knowledge-base-cloud-infrastructure

13.4.2 DATA SECURITY

Data provides a critical foundation for every operation of an organisation. Protecting and using it securely is central to a zero trust strategy.
Data security refers to the process of protecting data from unauthorized access and data corruption throughout its lifecycle.
Data security applies to a wide range of data storage and transfer situations such as personal files, email transfers, data stored in the cloud, secure web sites, digital devices, Internet of Things devices. Data security can be implemented by the use of **Encryption**, **passwords**

ENCRYPTION

Definition:

- This is the process of converting an original message (**plaintext**) into a form that cannot be used by unauthorised individuals (**ciphertext**).
- The main goal is to protect the information and communication from unauthorised users.
- It is derived from mathematical concepts and a set of rule-based calculations called **algorithms** to transform messages in ways that are hard to decode

- The original text is known as the **Plaintext**. The transformed text is known as **ciphertext**.
- The **Plaintext** is supplied to an **algorithm** and an **encryption key**, which create a ciphertext. This ciphertext can be **decrypted** with a valid key.
- An **Algorithm** is a program that contains a mathematical formula or method.
- The **Key** is the information used in conjunction with the algorithm to create the cipher text from the plaintext. It is a series of bits used in a mathematical algorithm. There are two types of keys: **Encryption** and **Decryption keys**.

Encryption Algorithms

1. (Triple) DES

Triple DES or 3DES is the successor to the DES algorithm, or the **Data Encryption Standard**.
DES is an obsolete symmetric-key method of data encryption. It was phased out because of its over-exploited vulnerabilities.
Originally, DES used to be a very popular encryption method. It is a creation of the engineers in IBM, from back in 1970. In 1977 it was adopted by the government of the USA as an encryption standard. DES was used for protecting unclassified but sensitive data.
It was the first algorithm approved by the federal government for public disclosure. It gained popularity fast among businesses dealing with confidential information – financial institutions, telecommunication companies, other software and Internet providers, etc.
DES encrypts data by grouping it into blocks and encoding them simultaneously by applying the key to the entire block – instead of bit by bit. The blocks are always 64-bit.
In DES, the same key is used to encipher and decipher the message. Decoding is nothing more than applying the same steps in reverse direction and reverse order.
The progress in technology, and more specifically in processing power made it possible for a DES-encrypted message to be brute force hacked in less than 24 hours.

3DES is one of the successors of DES. It was first made public in 1998 and is a simple variation of DES, where each block is encrypted via DES three times each. Triple DES uses three independent keys, with three separate 56-bit DES keys, or a total of 168 bits.
With its more complex way of working, the 3DES encryption method is slower. Still, the effectiveness of 3DES is considered to give enough security that it is approved to be used up until 2030. Triple DES is used for online payments, by apps in the Microsoft Office package, as well as by Mozilla's Firefox and Thunderbird.

2. RSA

The algorithm is named after its designers. In the 1970's Ron Rivest, Adi Shamir, and Leonard Adleman: Rivest-Shamir-Adleman came up with the encryption method while working for the Massachusetts Institute of Technology.
RSA is asymmetric, which means the system is not using the same key for decryption and encryption. The algorithm works via generating a public and a private key. The public and private key are not identical, but are linked to each other.
There is no difference which key is used for encryption and which is used for decryption. Both the private and the public key can be used to encode the message and the opposite key will have to be applied to decode it.
In fact, the difference is elsewhere: the public key, as its name suggests, can be introduced to the public, while the private key is the 'secret'.
The high level of security RSA provides is ensured by the difficulty of factoring long strings, the product of factoring two large prime numbers.
Both the private and the public keys are generated via multiplying two large numbers, creating a modulus. The modulus is used by both the public and the private key. The public key is created via the modulus, adding a public exponent, usually 65537. And the private key is created by the modulus, and a private exponent.
The computational difficulty of large integers makes RSA significantly safer than DES.
RSA is used for **SSH authentication**, for **SSL encryption**, and for protecting sensitive data in various browsers.

3. Blowfish

Blowfish is considered an alternative to the **DES** and **RSA** encryption methods. It was designed in 1993 by Bruce Schneier as a general-purpose algorithm. The security of the cipher has been tested and proven in time.
The technology of Blowfish is symmetric, which means one private key (or just key) is used to encrypt an entire block of data. The size of the blocks is 64-bit, and the length of the key may vary from a 32-bits to a 448-bit key. A full encryption has never been broken.
Blowfish is not patented. Its flexibility, speed and security gives it a competitive edge against its alternatives. It is can be used and explored free of charge. Therefore it is highly popular in cryptographic software.

4. Twofish

Twofish is a successor of Blowfish. It is a brain-child of the designer, Bruce Schneier, .
Twofish, uses block cyphering. It symmetric, which again means the same key is used for enciphering and deciphering. Twofish divides the message that needs encryption into 128-bit blocks and applies the key simultaneously to all blocks.
Twofish is considered to be fast, flexible, and at the same time, to have a conservative design. The encryption method has not been patented and is free to use.

5. AES

The **Advanced Encryption Standard** was designed by the Belgian cryptographers Vincent Rijmen and Joan Daemen back in 1998.

AES is symmetric encryption algorithm. It is considered to be more effective than its predecessors, DES and 3DES, using a more complex algorithm and a longer key. The decryption works faster, which makes it a better alternative for a cypher in routers, firewalls, security protocols and in general any applications that use encryption.
AES works with 128-bit blocks. Each cypher – the AES-128, -192, and -256 uses respectively 128-, 192-, and 256-bit keys.

6. IDEA

The International Data Encryption Algorithm (IDEA) was created by Xuejia Lai and James Massey and is another symmetric encryption algorithm, first described in 1991.
It uses 64-bit blocks and 128-bit keys. Like AES, IDEA uses rounds of encryption. Unlike AES, IDEA also adopts a methodology called half-rounds.
The IDEA algorithm has been patented and has only been available for free for non-commercial rules.

7. End-to-End Encryption

This is a system of communication where only the communicating users can read the messages. This prevents the third parties such as telecom provider and internet providers from being able to read the conversation.
In many messaging systems, data is encrypted while in transit. The messages are accessible to the service provider who store the message before the recipient retrieves it. This gives the third parties ability to search or scan for illegal and unacceptable content.
End-to-end encryption aims to prevent data from being read or modified other than by the true sender and recipient(s). The sender encrypts the messages and the third party does not have a means to decrypt them.

Asymmetric Algorithms
elliptic curve,

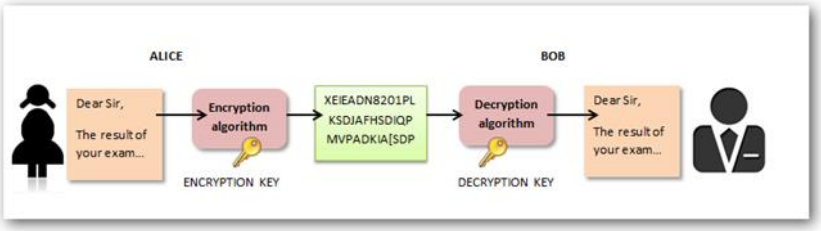
Diffie-Hellman),

Diffie Hellman (DH) key exchange algorithm is a method for securely exchanging cryptographic keys over a public communications channel. Keys are not actually exchanged – they are jointly derived. It is named after their inventors Whitfield Diffie and Martin Hellman

Types of Encryption

1. Symmetric Encryption

- The method requires the **same key** - a secret key- be used with the algorithm to both encrypt and decrypt the message.
- The sender and the receiver must find a way of exchanging the keys so that the process is not compromised. This can be done using different channels other than the one being used to transmit data.



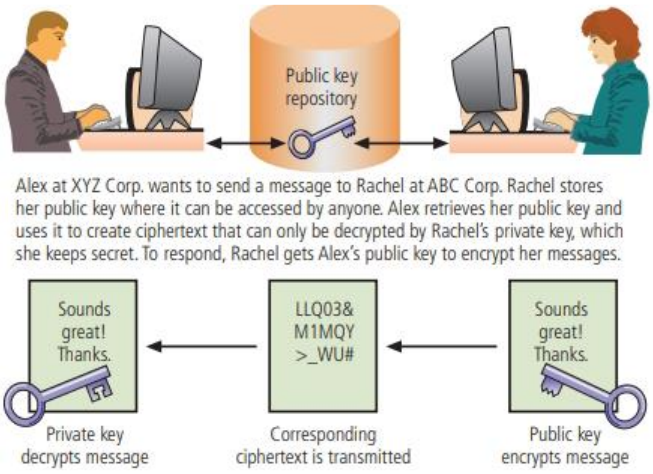
- Symmetric-key algorithms can be divided into **stream ciphers** and **block ciphers**—stream ciphers encrypt the bits of the message one at a time, and block ciphers take a number of bits, often in blocks of 64 bits at a time, and encrypt them as a single unit.

Symmetric Encryption Algorithms

- The most familiar is **Data Encryption Standard (DES).**
- DES was developed in 1977 by IBM and is based on the Data Encryption Algorithm (DEA), which uses a 64-bit block size and a 56-bit key. With a 56-bit key, the algorithm has 2S6 (more than 72 quadrillion) possible keys.
- **Triple DES (3DES)** was developed as an improvement to DES. It is substantially more secure than DES, not only because it uses as many as three keys instead of one, but because it performs three different encryption operations.
- The current industry and government standard in symmetric encryption is the **Advanced Encryption Standard (AES).** AES is based on the Rijndael Block Cipher, which features a variable block length and a key length of 128, 192, or 256 bits.

2. Asymmetric Encryption

- This is also known as **public key encryption**
- It uses two different keys to encrypt and decrypt a message.
- However, if Key A is used to encrypt the message, then only Key B can decrypt it; conversely, if Key B is used to encrypt a message, then only Key A can decrypt it. This technique is most valuable when one of the keys is private and the other is public.
- The public key is stored in a public location, where anyone can use it. The private key, as its name suggests, is a secret known only to the owner of the key pair
- The two keys are derived from complex mathematical calculations making it difficult to break the code.


Public key repository
Alex at XYZ Corp. wants to send a message to Rachel at ABC Corp. Rachel stores her public key where it can be accessed by anyone. Alex retrieves her public key and uses it to create ciphertext that can only be decrypted by Rachel's private key, which she keeps secret. To respond, Rachel gets Alex's public key to encrypt her messages.
Sounds great! Thanks.
LLQ03& M1MQY >_WU#
Sounds great! Thanks.
Private key decrypts message
Corresponding ciphertext is transmitted
Public key encrypts message

Symmetric and Asymmetric Encryption

Similarities

1. They both require a key to encrypt and decrypt the data. Symmetric uses a single key while the asymmetric uses two keys to encrypt and decrypt the data.
2. The result of the encryption process is a cipher text that needs to be converted back to plain text.

Differences

1. Keys

- Symmetric encryption algorithms make use of a single key. This key must be shared with the other person that the message was intended for. This compromise on the security of the data.
- Asymmetric encryption makes use of two different but related keys: The **Public key** and the **Private Key**. One uses a single key to encrypt while the other key is used to decrypt the message.

2. **Secrecy of the Key**
   - In symmetric encryption the key must be kept secret and only available to the person whom you intent to share the message with
   - In asymmetric systems, the key used for encryption is known as the public key and can be freely shared with others. On the other hand, the key used for decryption is the private key and should be kept in secret.
3. **Key Length**
   - In symmetric schemes, the keys are randomly selected, and their lengths are usually set at 128 or 256 bits, depending on the required level of security.
   - In asymmetric encryption, however, there must be a mathematical relationship between the public and private keys, meaning that there is a mathematical pattern between the two.
   - Due to the fact that this pattern can potentially be exploited by attackers to crack the encryption, asymmetric keys need to be much longer to present an equivalent level of security. The difference in key length is so pronounced that a 128-bit symmetric key and a 2,048-bit asymmetric key offer roughly similar levels of security.
4. **Processing**
   - Symmetric encryption algorithms are much faster and require less computational power.
   - Asymmetric encryption solves the problem of key distribution by using public keys for encryption and private keys for decryption. The trade-off, however, is that asymmetric encryption systems are very slow by comparison to symmetric systems and require much more computing power as a result of their vastly longer key lengths.

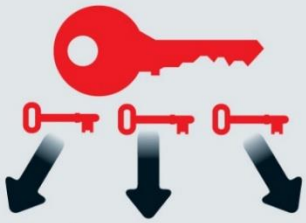
https://www.youtube.com/watch?v=AQDCe585Lnc

How data encryption works

HOW IT WORKS

Public-key encryption uses a combination of cryptographic keys to keep your data safe

1 Public-private key pair
Your computer owns two unique cryptographic keys that are mathematically related to each other: one is public, while one is completely private.



2 Key hand-out
It makes your public key freely available to any other computer that wants to send a file to you.

3 Encryption
When someone sends you a file, their computer encrypts it with a symmetric key, meaning it can only be opened with the identical key.

4 Public key encryption
Both the encrypted file and the key needed to decode it are placed in a file and encrypted using the public key.

5 File transfer
The public-key encrypted file is sent to your computer.

6 Decryption step 1
Files encrypted with your public key can only be decrypted by your private key; even if the message is intercepted by another computer that knows your public key, only your computer can decrypt its contents.

7 Extraction
Your computer extracts the encrypted file and the symmetric key needed to decode it.

8 Decryption step 2
The computer decodes the file using the symmetric key.

9 File unlock
Voila! The contents of the file are now yours to view.

encryption approaches; e.g. file/data block level, folder level, full disk

Password

- This is a memorised secret text used to confirm the identity of a user. The person who memorise the password is called the **claimant** while the party verifying the identity of the claimant is called the **verifier**.
- When the claimant successfully demonstrates knowledge of the password to the verifier through an established authentication protocol the verifier is able to infer the claimant's identity.
- a password is an arbitrary string of characters including letters, digits, or other symbols

How Passwords are stored

1. Plain Text Passwords

- The simplest way a site can store your password is in plain text.
- This means somewhere on a server, there exists a database with your username and password in it in a human-readable form, when you enter your credentials on the site, it checks them against the database to see if they match.

2. Password Encryption

- Most sites encrypt password before they store it on their servers.
- Encryption uses a special key to turn your password into a random string of text.
- If a hacker were to get hold of this random string of text, they wouldn't be able to log into your account unless they also had the key, which they could then use to *decrypt* it.

3. Hashed Passwords

- Hashed is similar to encryption in the sense that it turns your password into a long string of letters and numbers to keep it hidden.
- However, unlike encryption, hashing is a one way street: If you have the hash, you can't run the algorithm backwards to get the original password. This means a hacker would have to obtain the hashes and then try a number of different password combinations to see which ones worked.
- However, there is a downside to this method. While a hacker can't decode a hash back to the original password, they *can* try many different passwords until one matches the hash they have.
- Computers can do this very fast, and with the help of something called rainbow tables—which is essentially a list of trillions of different hashes and their matching passwords—they can just look up the hash to see if it's already been discovered.

4. Hashed Passwords with a Dash of Salt

- Salting a hash means adding a random string of characters—called a "salt"—to the beginning or end of your password before hashing it.
- It uses a different salt for each password, and even if the salts are stored on the same servers, it will make it very hard to find those salted hashes in the rainbow tables, since each one is long, complex, and unique.

https://lifehacker.com/how-your-passwords-are-stored-on-the-internet-and-when-5919918

13.5.1 CERTIFICATE BASED ENCRYPTION

Digital Certificate

- A digital signature is basically a way to ensure that an electronic document (e-mail, spreadsheet, text file, etc.) is authentic. Authentic means that you know who created the document and you know that it has not been altered in any way since that person created it.
- Digital signatures rely on certain types of encryption to ensure authentication. Encryption is the process of taking all the data that one computer is sending to another and encoding it into a form that only the other computer will be able to decode. Authentication is the process of verifying that information is coming from a trusted source. These two processes work hand in hand for digital signatures.
- A Digital Certificate is issued by a trusted **third party**, which proves sender's identity to the receiver and receiver's identity to the sender.
- It consists of a **public key** plus a **user ID** of the key owner, with the whole block signed by a trusted third party called the **Certificate Authority (CA)**. The certificate also includes some information about the third party plus an indication of the period of validity of the certificate

Digital certificate contains-

1. Name of certificate holder.
2. Serial number which is used to uniquely identify a certificate, the individual or the entity identified by the certificate
3. Expiration dates.
4. Copy of certificate holder's public key.(used for decrypting messages and digital signatures)
5. Digital Signature of the certificate issuing authority.

How the Certificate is created?

The key steps can be summarised as follows:

1. User software (client) creates a pair of keys: one public and one private.
2. Client prepares an unsigned certificate that includes the user ID and user's public key.
3. User provides the unsigned certificate to a CA in some secure manner. This might require a face-to-face meeting, the use of registered e-mail, or happen via a Web form with e-mail verification.
4. CA creates a signature as follows:
    a. CA uses a hash function to calculate the hash code of the unsigned certificate.
    A hash function is one that maps a variable-length data block or message into a fixed-length value called a hash code, such as SHA family
    b. CA generates digital signature using the CA's private key and a signature generation algorithm.
5. CA attaches the signature to the unsigned certificate to create a signed certificate.
6. CA returns the signed certificate to client.
7. Client may provide the signed certificate to any other user.
8. Any user may verify that the certificate is valid as follows:
    a. User calculates the hash code of certificate (not including signature).
    b. User verifies digital signature using CA's public key and the signature verification algorithm. The algorithm returns a result of either signature valid or invalid.

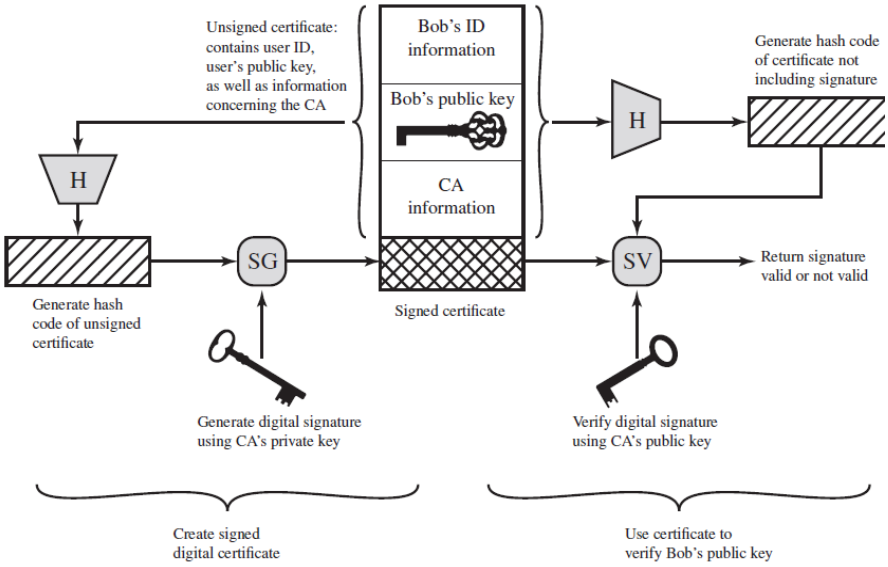

Unsigned certificate: contains user ID, user's public key, as well as information concerning the CA
Bob's ID information
Bob's public key
CA information
Generate hash code of certificate not including signature
H
H
SG
SV
Return signature valid or not valid
Generate hash code of unsigned certificate
Signed certificate
Generate digital signature using CA's private key
Verify digital signature using CA's public key
Create signed digital certificate
Use certificate to verify Bob's public key

Figure 2.8 **Public-Key Certificate Use**

- One scheme has become universally accepted for formatting public-key certificates:
  - **The X.509 standard.** X.509 certificates are used in most network security applications, including IP Security (IPsec), Transport Layer Security (TLS), SecureShell (SSH), and Secure/Multipurpose Internet Mail Extension (S/MIME).

- A **digital signature** is basically a way to ensure that an electronic document (e-mail, spreadsheet, text file, etc.) is **authentic**.
- Digital signatures rely on certain types of encryption to ensure authentication.

https://www.youtube.com/watch?v=E89gTsSEZOo
https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/security/encryption/
https://www.wikihow.com/Be-Your-Own-Certificate-Authority
https://www.youtube.com/watch?v=jSjK9BuURS0
https://www.geeksforgeeks.org/digital-signatures-certificates/
https://www.youtube.com/watch?v=stsWa9A3sOM
https://www.youtube.com/watch?v=q9vu6_2r0o4

13.4.3 DATABASE MANAGEMENT SYSTEMS

Database

- A database is a collection of related data that represents some aspect of the real world. A database system is designed to be built and populated with data for a certain task.

DBMS

- Database Management System (DBMS) is a software for storing and retrieving users' data
- It consists of a group of programs that manipulate the database.
- The DBMS accepts the request for data from an application and instructs the operating system to provide the specific data.

Characteristics of Database Management System

- Provides data security
- Minimise data redundancy
- Insulation between programs and data abstraction
- Support of multiple views of the data
- Sharing of data and multiuser transaction processing
- DBMS allows entities and relations among them to form tables.
- It follows the **ACID** concept (**Atomicity**, **Consistency**, **Isolation**, and **Durability**).

RDBMS vs. Flat File

DBMS	Flat File Management System
Multi-user access	It does not support multi-user access
Design to fulfil the need for small and large businesses	It is only limited to smaller DBMS system.
Remove redundancy and Integrity	Redundancy and Integrity issues
Expensive. But in the long term Total Cost of Ownership is cheap	It's cheaper
Easy to implement complicated transactions	No support for complicated transactions

Popular DBMS Software

- MySQL, Microsoft Access, Oracle, PostgreSQL, dBASE, FoxPro, SQLite, IBM DB2, LibreOffice Base, MariaDB, Microsoft SQL Server

Application of DBMS

Sector	Use of DBMS
Banking	For customer information, account activities, payments, deposits, loans, etc.
Airlines	For reservations and schedule information.
Universities	For student information, course registrations, colleges and grades.
Telecommunication	It helps to keep call records, monthly bills, maintaining balances, etc.
Finance	For storing information about stock, sales, and purchases of financial instruments like stocks and bonds.
Sales	Use for storing customer, product & sales information.
Manufacturing	It is used for the management of supply chain and for tracking production of items. Inventories status in warehouses.
HR Management	For information about employees, salaries, payroll, deduction, generation of paychecks, etc.

Types of DBMS

Hierarchical DBMS

- Data is organised in a tree-like structure.
- Data is Stored Hierarchically (top down or bottom up) format.
- Data is represented using a parent-child relationship.
- Parent object may have many children, but children have only one parent.

Network Model

- The network database model allows each child to have multiple parents.
- It helps you to address the need to model more complex relationships like as the orders/parts many-to-many relationship.
- In this model, entities are organised in a graph which can be accessed through several paths.

Relational model

- Relational DBMS is the most widely used DBMS model because it is one of the easiest.
- This model is based on normalising data in the rows and columns of the tables.
- Relational model stored in fixed structures and manipulated using SQL.

Object-Oriented Model

- In Object-oriented Model data stored in the form of objects.
- The structure which is called classes which display data within it.
- It defines a database as a collection of objects which stores both data members values and operations.

Advantages of DBMS

- DBMS offers a variety of techniques to store & retrieve data
- DBMS serves as an efficient handler to balance the needs of multiple applications using the same data
- Uniform administration procedures for data
- Application programmers never exposed to details of data representation and storage.
- A DBMS uses various powerful functions to store and retrieve data efficiently.
- Offers Data Integrity and Security
- The DBMS implies integrity constraints to get a high level of protection against prohibited access to data.
- A DBMS schedules concurrent access to the data in such a manner that only one user can access the same data at a time
- Reduced Application Development Time

Disadvantage of RDBMS

- Cost of Hardware and Software of a DBMS is quite high which increases the budget of your organisation.
- Most database management systems are often complex systems, so the training for users to use the DBMS is required.
- In some organisations, all data is integrated into a single database which can be damaged because of electric failure or database is corrupted on the storage media
- Use of the same program at a time by many users sometimes lead to the loss of some data.
- DBMS can't perform sophisticated calculations

DBMS Functions

1. Data Dictionary Management

Data Dictionary is where the DBMS stores definitions of the data elements and their relationships (metadata). The DBMS uses this function to look up the required data component structures and relationships. When programs access data in a database they are basically going through the DBMS. This function removes structural and data dependency and provides the user with data abstraction. In turn, this makes things a lot easier on the end user. The Data Dictionary is often hidden from the user and is used by Database Administrators and Programmers.

2. Data Storage Management

This particular function is used for the storage of data and any related data entry forms or screen definitions, report definitions, data validation rules, procedural code, and structures that can handle video and picture formats. Users do not need to know how data is stored or manipulated. Also involved with this structure is a term called performance tuning that relates to a database's efficiency in relation to storage and access speed.

3. Data Transformation and Presentation

This function exists to transform any data entered into required data structures. By using the data transformation and presentation function the DBMS can determine the difference between logical and physical data formats.

4. Security Management

This is one of the most important functions in the DBMS. Security management sets rules that determine specific users that are allowed to access the database. Users are given a username and password or sometimes through biometric authentication (such as a fingerprint or retina scan) but these types of authentication tend to be more costly. This function also sets restraints on what specific data any user can see or manage.

5. Multiuser Access Control

Data integrity and data consistency are the basis of this function. Multiuser access control is a very useful tool in a DBMS, it enables multiple users to access the database simultaneously without affecting the integrity of the database.

6. Backup and Recovery Management

Backup and recovery is brought to mind whenever there is potential outside threats to a database. For example if there is a power outage, recovery management is how long it takes to recover the database after the outage. Backup management refers to the data safety and integrity; for example backing up all your mp3 files on a disk.

7. Data Integrity Management

The DBMS enforces these rules to reduce things such as data redundancy, which is when data is stored in more than one place unnecessarily, and maximising data consistency, making sure database is returning correct/same answer each time for same question asked.

8. Database Access Languages and Application Programming Interfaces

A query language is a nonprocedural language. An example of this is SQL (structured query language). SQL is the most common query language supported by the majority of DBMS vendors. The use of this language makes it easy for user to specify what they want done without the headache of explaining how to specifically do it.

9. Database Communication Interfaces

This refers to how a DBMS can accept different end user requests through different network environments. An example of this can be easily related to the internet. A DBMS can provide

access to the database using the Internet through Web Browsers (Mozilla Firefox, Internet Explorer, and Chrome).

10. Transaction Management

This refers to how a DBMS must supply a method that will guarantee that all the updates in a given transaction are made or not made. All transactions must follow what is called the ACID properties.

Properties of a transaction (ACID)

A – Atomicity: states a transaction is an indivisible unit that is either performed as a whole and not by its parts, or not performed at all. It is the responsibility of recovery management to make sure this takes place.

C – Consistency: A transaction must alter the database from one constant state to another constant state.

I – Isolation: Transactions must be executed independently of one another. Part of a transaction in progress should not be able to be seen by another transaction.

D – Durability: A successfully completed transaction is recorded permanently in the database and must not be lost due to failures**.**

Access Control

- Access control is a security technique that regulates who or what can view or use resources in a computing environment. It is a fundamental concept in security that minimises risk to the business or organisation.
- There are two types of access control: **physical** and **logical**.
- Physical access control limits access to campuses, buildings, rooms and physical IT assets.
- Logical access control limits connections to computer networks, system files and data.
- Access control systems perform **identification, authentication** and of users and entities by evaluating required login credentials that can include **passwords**, **personal identification numbers (PINs), biometric scans**, **security tokens** or other authentication factors.
- These security controls work by identifying an individual or entity, verifying that the person or application is who or what it claims to be **(authentication),** and authorising the access level and set of actions associated with the username or IP address (**authorisation**)
- Organisations use different access control models depending on their compliance requirements and the security levels of information technology they are trying to protect.

Types of access control

- **Mandatory access control (MAC)**: A security model in which access rights are regulated by a central authority based on multiple levels of security.
- Classifications are assigned to system resources and the operating system grants or denies access to those resource objects based on the information security clearance of the user or device.
- **Discretionary access control (DAC):** An access control method in which owners or administrators of the protected system, data or resource set the policies defining who or what is authorised to access the resource. Many of these systems enable administrators to limit the propagation of access rights.
- **Role-based access control (RBAC):** A widely used access control mechanism that restricts access to computer resources based on individuals or groups with defined business functions - executive level, engineer level 1 - rather than the identities of individual users. The role-based security model relies on a complex structure of role assignments, role authorisations and role permissions developed using role engineering to regulate employee access to systems.
- **Rule-based access control:** A security model in which the system administrator defines the rules that to govern access to resource objects. Often these rules are based on conditions, such as time of day or location.

- **Attribute-based access control (ABAC):** A methodology that manages access rights by evaluating a set of rules, policies and relationships using the attributes of users, systems and environmental conditions.

Use of Access Control

- The goal of access control is to minimise the risk of unauthorised access to physical and logical systems.
- Access control is a fundamental component of security compliance programs that ensures security technology and access control policies are in place to protect confidential information, such as customer data.
- Access control systems are complex and can be challenging to manage in dynamic IT environments that involve on-premises systems and cloud services.

Access Control Methods

- Refers to method by which a user is granted or denied access to a resource.
- The most common methods are
    - Username and Password
    - Security Policy
    - Grant / Revoke Privileges

Database View

- This is a subset of a database and is based on a query that runs on one or more database tables.
- Database views are saved in the database as named queries and can be used to save frequently used, complex queries.
- A View contains no data of its own but it is like window through which data from tables can be viewed or changed.
- The table on which a View is based are called **BASE** Tables.
- Database views are used to restrict data access.
- There are different types of database views: **Dynamic, Static, Simple** and **Complex Views**
- **Dynamic views**
    - Can contain data **from one or two tables** and automatically **include all of the columns** from the specified table or tables.
    - Are automatically updated when related objects or extended objects are created or changed.
- **Static views**
    - Can contain **data from multiple tables** and the r**equired columns** from these tables must be specified in the **SELECT** and **WHERE** clauses of the static view.
    - Must be manually updated when related objects or extended objects are created or changed.
- **Simple views**
    - They can only contain a single base table.
- **Complex views**
    - They can be constructed on more than one base table. In particular, complex views can contain: join conditions, a group by clause, a order by clause.

The key differences between these types of Views are:

Simple View	Complex View
Contains only one single base table or is created from only one table.	Contains more than one base tables or is created from more than one tables.

We cannot use group functions like MAX(), COUNT(), etc.	We can use group functions.
Does not contain groups of data.	It can contain groups of data.
INSERT, DELETE and UPDATE are directly possible on a simple view.	We cannot apply INSERT, DELETE and UPDATE on complex view directly.
Does not include NOT NULL columns from base tables.	NOT NULL columns that are not selected by simple view can be included in complex view.

The reasons for creating view:

1) When data security is required.
2) When data redundancy is to be kept to the minimum while maintaining data security.

14.0 USING ‘IT SYSTEMS’ IN ORGANISATIONS

Introduction

- We are surrounded by networks through which information flows constantly
- Our notions of time and location are changing – the world seems to have become a **'global village'** where distance is no longer a barrier to commercial or social contact.
- It is difficult to imagine being without all the networked infrastructure that plays a crucial part in our daily lives.

- **IT system refers to :** *all computer systems, programs, networks, hardware, intellectual property software, software engines, databases, operating systems, Internet websites, Website content and links and equipment used to process, store, maintain and operate data, information and functions used in, intended to be used in, or held for use in, the business of the Company or the Acquired Subsidiaries.*

Role of IT in an Organisation

- **Operational Support**;
    - These activities are used to manage the day-to-day activities of an organisation. These activities include monitoring and controlling the production lines, analysing the sales, managing/maintaining inventory, configuring network components, allocating services, managing faults.
- **Collaboration;**
    - The collaboration role allows two or more staff to work on the same assignment at the same time. It also involve communication between workers so that they can fulfil organisation’s objective. Use of IT in collaboration activities include communication tools such as email or video conferencing, project management, team working
- **Knowledge Management**; e.g. storing and retrieving knowledge, finding sources, mining data for knowledge, content/document management, sharing knowledge.
- **Product Development**; e.g. systems for research, planning, design, prototyping, marketing, testing
- **Service Delivery**; e.g. enterprise service management (ESM) software, design of processes to deliver services to the organisation and its customers, workflow management.

What Is an Information System?

- An information system (IS) is a set of components that work together to manage data processing and storage. Its role is to support the key aspects of running an organization, such as communication, record keeping, decision-making, data analysis and more.
- Companies use this information to improve their business operations, make strategic decisions and gain a competitive edge.
- Information technology has an essential role in business's daily operations and success.
- The role of IT in business is seen in how it can help a company become more productive, increase performance, save money, improve the customer experience, streamline communications and enhance managerial decision-making.
- It also plays a role in helping companies expand globally and in providing staff access to company information wherever and whenever they need it.

Role of IT in an organisation

Communication between Employees, Suppliers and Customers

- For many companies, email is the principal means of communication between employees, suppliers and customers. Email was one of the early drivers of the Internet, providing a simple and inexpensive means to communicate.
- Other communications tools have also evolved, allowing staff to communicate using live chat systems, online meeting tools and video-conferencing systems.
- Voice over internet protocol (VOIP) telephones and smart-phones offer even more high-tech ways for employees to communicate.
- Video conferencing software allow users to exchange video in real time.

Inventory Management Systems

- Inventory management systems track the quantity of each item a company maintains, triggering an order of additional stock when the quantities fall below a pre-determined amount.
- These systems are best used when the inventory management system is connected to the **point-of-sale** (POS) system. The POS system ensures that each time an item is sold, one of that item is removed from the inventory count, creating a closed information loop between all departments.

Data Management Systems

- Today, most companies store digital versions of documents on servers and storage devices.
- The documents become instantly available to everyone in the company, regardless of their geographical location.
- Companies are able to store and maintain a tremendous amount of historical data economically, and employees benefit from immediate access to the documents they need.

Management Information Systems

- Storing data is only a benefit if that data can be used effectively. Progressive companies use that data as part of their strategic planning process as well as the tactical execution of that strategy.
- Management Information Systems (MIS) enable companies to track sales data, expenses and productivity levels. The information can be used to track profitability over time, maximise return on investment and identify areas of improvement.
- Managers can track sales on a daily basis, allowing them to immediately react to lower-than-expected numbers by boosting employee productivity or reducing the cost of an item.
- Schools also use a software that stores all the records about a student. When a teacher wants to track the progress of the student he just access the system that displays the student records.

Customer Relationship Management/ Customer Experience

- Companies are using IT to improve the way they design and manage customer relationships.

- **Customer Relationship Management (**CRM) systems capture every interaction a company has with a customer, so that a more enriching experience is possible.
- If a customer calls a call centre with an issue, the customer support representative will be able to see what the customer has purchased, view shipping information, call up the training manual for that item and effectively respond to the issue.
- The entire interaction is stored in the CRM system, ready to be recalled if the customer calls again.
- The customer has a better, more focused experience and the company benefits from improved productivity.

https://bizfluent.com/about-6744256-importance-information-technology-business-sector.html

14.1.2

TRANSACTION PROCESSING SYSTEM

- A transaction process system (TPS) is an information processing system for business transactions involving the **collection**, **modification** and **retrieval** of all transaction data
- A Transaction is an event that **generates** or **modifies** data to be stored in an information system

Types of TP

- **Batch processing:**
    - Processes several transactions at the same time, with a time delay.
    - Does not require user interaction during processing.
    - Results not available to influence the next action
- **Real-time processing:**
    - Deals with one transaction at a time and does not have a time delay.
    - Results are immediately available and can influence the next activity

Features/Characteristics of TP

- **Rapid Response:** The response time of a transaction processing system (TPS) is important because a business cannot afford to have their customers waiting for long periods of time before making a transaction.
- **Reliability**: A good TPS must be very reliable because if it were to break down businesses could lose a huge portion of revenue because customers would not be able to purchase their products.
- **Inflexibility**: The TPS must work the same way for every transaction as long as the TPS is being used. The formality and structure should never change.
- **Controlled processing**: The TPS must be able to allow authorised employees to be able to access it at any time.
- User transparency/ease of use,
- **Scalability**: This is the property of a system to handle a growing amount of work by adding resources to the system. In an economic context, a scalable business model implies that a company can increase sales given increased resources. As transaction increases more resources can be added to the system without affecting its performance

Uses of Transaction Processing System

- Sales and orders in supermarkets and large stores
- Payments
- Delivery tracking,
- Process tracking,
  People tracking (such as passports/ID cards at borders, automatic number plate reading on roads, location awareness on mobile devices),
  Financial such as payroll, banks
- Personnel/student records – in schools, Universities, Colleges

Why Use Transaction Processing Systems

- Cost savings- No need to employ several people to manage the process such as cashers

- Efficiency-
- Better access to customers/markets,
- Automated data collection – Reward system such as bonga Points
- Better user experience.

14.1.3 CUSTOMER RELATIONSHIP MANAGEMENT SYSTEM

Summary

Types of CRM

- **Operational**; Marketing, Sales, Customer Service
- **Analytical**; Data Management, Finding And Retaining Customers
- **Collaborative**; Improving Communications, Tracking And Sharing Data.

Features of CRM; Automation, Data Collection and Analysis, Management

How CRM Is Used, Roles

- Identifying Sales Prospects/Leads, Tracking Customer Interactions, Filtering Enquiries to Contact/Call Centres, Providing Location Awareness, Tracking Employee Performance, Improving Customer Satisfaction/Experience, Tracking And Engaging With Social Media, Improving Business-To-Business Links

Why CRM is used;

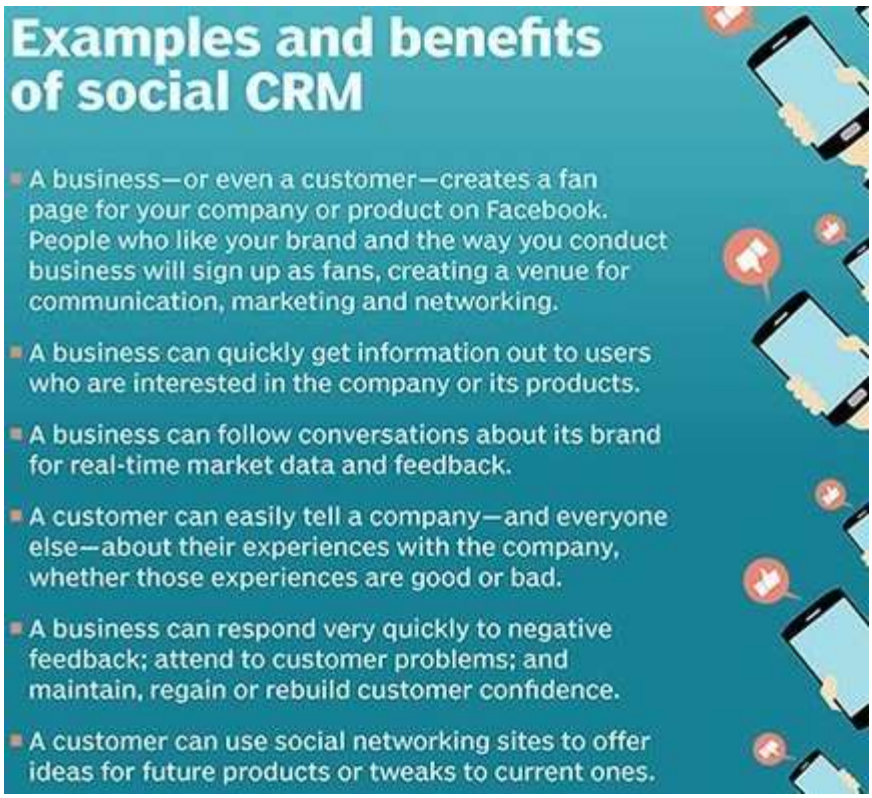
- Improving Efficiency, Saving Money, Improved Management, Improved Communication Within The Organisation, Improved/More Accurate Data Management.

CUSTOMER RELATIONSHIP MANAGEMENT SYSTEM

- Customer relationship management (CRM) is the combination of practices, strategies and technologies that companies use to manage and analyse customer interactions and data throughout the customer lifecycle, with the goal of improving customer service relationships and assisting in customer retention and driving sales growth.
- CRM systems compile customer data across different channels, or points of contact between the customer and the company, which could include the company's website, telephone, live chat, direct mail, marketing materials and social media.
- Businesses can use CRM software to keep an up-to-date list of contacts, manage leads, help forecast sales and track the company's social media sites.
- CRM can also communicate through email and instant messaging with customers and staff.
- It can also be used to manage marketing campaigns and create customised dashboards showcasing sales analytics and customer activity.
- CRM systems can also give customer-facing staff detailed information on customers' personal information, purchase history, buying preferences and concerns.

Components of CRM

- **Marketing automation**: CRM tools with marketing automation capabilities can automate repetitive tasks to enhance marketing efforts at different points in the lifecycle. For example, as sales prospects come into the system, it might automatically send the prospects marketing materials, typically via email or social media, with the goal of turning a sales lead into a full-fledged customer.
- **Sales force automation**: Sales force automation tools track customer interactions and automate certain business functions of the sales cycle that are necessary to follow leads and attract and obtain new customers.
- **Contact centre automation**: Designed to reduce tedious aspects of a contact centre agent's job, contact centre automation might include pre-recorded audio that assists in customer problem-solving and information dissemination. Various software tools that integrate with the agent's desktop tools can handle customer requests in order to cut down on the time of calls and to simplify customer service processes.
- **Geolocation technology**, or location-based services: Some CRM systems include technology that can create geographic marketing campaigns based on customers' physical locations, sometimes integrating with popular location-based GPS apps. Geolocation technology can also be used as a networking or contact management tool in order to find sales prospects based on a location.


Examples and benefits of social CRM
A business—or even a customer—creates a fan page for your company or product on Facebook. People who like your brand and the way you conduct business will sign up as fans, creating a venue for communication, marketing and networking.
A business can quickly get information out to users who are interested in the company or its products.
A business can follow conversations about its brand for real-time market data and feedback.
A customer can easily tell a company—and everyone else—about their experiences with the company, whether those experiences are good or bad.
A business can respond very quickly to negative feedback; attend to customer problems; and maintain, regain or rebuild customer confidence.
A customer can use social networking sites to offer ideas for future products or tweaks to current ones.

14.1.4 MANAGEMENT INFORMATION SYSTEMS

- It is a system of hardware and software used for business-critical decision-making within an organisation
- It refers to computer software that is used to store, organise and analyse information.
- Management information systems are used to track **sales**, **inventory**, **equipment** and **related business information**.

TYPES OF MIS

- **Transaction Processing System (TPS**), which processes the routine transactions associated with a business. Example transactions include payroll processing, order processing, such as for an e-commerce business, and invoicing.
- **Management Support Systems (MSS),** which store and organise data, enabling end users to generate reports and analyse data to address business needs and inform planning. A data warehouse is an example of a Management Support System.
- **Decision Support Systems (**DSS**)**, which analyse business data to assist managers with decision making. For example, a DSS could project revenue figures based on new product sales assumptions.
- **Expert Systems**, which provide managers with insights and advice, using artificial intelligence (AI) to simulate the expert knowledge of a human in a particular field.

14.1.5. INTELLIGENT TRANSPORT SYSTEMS

• **Advantages**; e.g. pedestrian safety, pollution reduction, accident prevention, cost savings, increased capacity, increased reliability

• **Disadvantages**; e.g. can emphasise one method of transport/transport Company to the disadvantage of another causing effects such as job loss or dissatisfaction, over management of people/situations that are already performing well, people becoming over-reliant on the system

• **Uses**; e.g. town planning, smart cities, public transport management/coordination, traffic management such as changing traffic lights or speed limits in response to traffic flow, traveller information systems, intra and inter vehicle systems such as (self- driving vehicles, collision avoidance and other driver aids), commercial vehicle scheduling and tracking, travel payment systems.

Definition

- It is the application of **sensing**, **analysis**, **control** and **communications technologies** to **ground transportation** in order to improve **safety**, **mobility** and **efficiency**.
- It includes a wide range of applications that process and share information to ease congestion, improve traffic management, minimise environmental impact and increase the benefits of transportation to commercial users and the public in general
- It is part of the Internet of Things incorporates both wireless and wire line communications-based information and electronics technologies.
- Wireless technology is used to connect vehicle information and location to other vehicles, other transportation modes (such as pedestrians or bicyclists), local infrastructure and remote infrastructure in the cloud.
- It is having a significant effect on transportation in applications such as electronic toll collection, ramp meters, traffic light cameras, traffic-signal coordination, transit signal priority and traveller-information systems.
- It can be used in
    - Fleet Monitoring,
    - Tolling Management,
    - Ticket Management,
    - Transportation Pricing
    - Traffic Monitoring.

- **Beneficiaries**
    - safety improvements -
    - availability of real-time information on a vehicle and analytics of travellers, businesses and transportation agencies.

14.1.6 EXPERT SYSTEMS

- **Def**:
    - An Expert System is computer-based decision-making system which uses both facts and heuristics to solve complex decision-making problems.
- It is mainly used to give advice to users in a specific field or knowledge domain
- It uses knowledge that has been acquired from different experts to reach a decision
- Expert systems were the predecessor of the current day artificial intelligence, deep learning and machine learning systems.

Examples of Expert Systems

- **MYCIN:** It was based on backward chaining and could identify various bacteria that could cause acute infections. It could also recommend drugs based on the patient's weight.
- **DENDRAL:** Expert system used for chemical analysis to predict molecular structure.
- **PXDES:** Expert system used to predict the degree and type of lung cancer
- **CaDet:** Expert system that could identify cancer at early stages

Components of the expert system

Components of an Expert System

- **User Interface**
  - It is an interface that helps the user communicate with the expert system.
  - It takes the user's query in a readable form and passes it to the inference engine.
  - After that, it displays the results to the user. In other words,

Inference Engine

- The inference engine is the brain of the expert system.
- It contains rules to solve a specific problem.
- It refers the knowledge from the Knowledge Base.
- It selects facts and rules to apply when trying to answer the user's query.
- It provides reasoning about the information in the knowledge base.
- It also helps in deducting the problem to find the solution.
- This component is also helpful for formulating conclusions.

Knowledge Base

- This is the repository of facts. It stores all the knowledge about the problem domain.
- It is like a large container of knowledge which is obtained from different experts of a specific field.

Facts and Rules

- A fact is a small portion of important information. Facts on their own are of very limited use. The rules are essential to select and apply facts to a user problem.

Knowledge Acquisition

- The term knowledge acquisition means how to get required domain knowledge by the expert system. The entire process starts by extracting knowledge from a human expert, converting the acquired knowledge into rules and injecting the developed rules into the knowledge base.

Knowledge Extraction Process

The process of Building an Expert Systems

- Determining the characteristics of the problem
- Knowledge engineer and domain expert work in coherence to define the problem
- The knowledge engineer translates the knowledge into a computer-understandable language. He designs an inference engine, a reasoning structure, which can use knowledge when needed.
- Knowledge Expert also determines how to integrate the use of uncertain knowledge in the reasoning process and what type of explanation would be useful.

Conventional System vs. Expert system

Conventional System	Expert System
Knowledge and processing are combined in one unit.	Knowledge database and the processing mechanism are two separate components.
The programme does not make errors (Unless error in programming).	The Expert System may make a mistake.
The system is operational only when fully developed.	The expert system is optimised on an ongoing basis and can be launched with a small number of rules.
Step by step execution according to fixed algorithms is required.	Execution is done logically & heuristically.
It needs full information.	It can be functional with sufficient or insufficient information.

Differences : Human expert vs. expert system

Human Expert	Artificial Expertise
Perishable	Permanent
Difficult to Transfer knowledge from	Transferable
Difficult to Document	Easy to Document
Unpredictable	Consistent
Expensive	Cost effective System

Benefits of expert systems

- It improves the decision quality
- Cuts the expense of consulting experts for problem-solving
- It provides fast and efficient solutions to problems in a narrow area of specialisation.
- It can gather scarce expertise and used it efficiently
- Offers consistent answer for the repetitive problem
- Helps you to get fast and accurate answers
- A proper explanation of decision making
- Ability to solve complex and challenging issues
- Expert Systems can work steadily work without getting emotional, tensed or fatigued.

Limitations of the expert system

- Unable to make a creative response in an extraordinary situation
- Errors in the knowledge base can lead to wrong decision
- The maintenance cost of an expert system is too expensive
- Each problem is different therefore the solution from a human expert can also be different and more creative

Applications of expert systems

- Information management
- Hospitals and medical facilities – Giving patients advice or diagnosis of diseases
- Help desks management -
- Employee performance evaluation
- Loan analysis -
- Virus detection
- Useful for repair and maintenance projects
- Financial decision making Knowledge publishing
- Process monitoring and control

- Supervise the operation of the plant and controller
- Stock market trading
- Airline scheduling & cargo schedules

IT GOVERNANCE AND POLICY

Definition

- Governance is “the set of responsibilities and practices exercised by the board and executive management with the goal of providing strategic direction, ensuring that objectives are achieved, ascertaining that risks are managed appropriately, and verifying that the enterprise’s resources are used responsibly”
- Governance is a process by which a board of directors, through management, guides an institution in fulfilling its corporate mission and protects the institution’s assets.
- Effective governance occurs when a board provides proper guidance to management regarding the strategic direction for the institution and oversees management’s efforts to move in this direction
- IT governance has been necessitated by a high profile corporate frauds and deception cases
- IT governance provides a structure for aligning IT strategy with business strategy. By following a formal framework, organisations can produce measurable results toward achieving their strategies and goals.
- A formal program also takes stakeholders' interests into account, as well as the needs of staff and the processes they follow. In the big picture, IT governance is an integral part of overall enterprise governance.
- IT governance is a formal framework that provides a structure for organisations to ensure that IT investments support business objectives.
- IT governance is the method by which decisions are made around IT investments. Establishing IT governance in an organisation is an important way of ensuring that IT policies and procedures are adhered to.
- IT governance is mainly carried out through establishment of policies.
- IT governance is a part of corporate governance and that businesses/organisations develop IT governance policies to comply with:
    - **national and international law -** Organisation wants to ensure that they are following the laid down rules of the country they are operating in. With information becoming a commodity, and personal data becoming an issue it is important the organisation have guidelines of how to deal with this information without breaking the law
    - **industry regulations and codes** – Each industry has regulations that safeguard it and ensure an even playing field. These guidelines are agreed upon by the players and need to be adhered to
    - **Organisation’s own internal needs**- there must be order within an organisation. Polices lay down guidelines what is required of an employee when within and outside his/her work area and how the employee handle organisation resources.

Why IT Governance

- Reducing corporate IT failures -
- preventing fraud and other financial crimes
- Providing oversight on how IT is being used by the organisation to fulfil its strategic goal
- Imposing ethical codes
- Protection of confidential information
- Financial accountability
- Data retention
- Disaster recovery

IMPORTANCE OF POLICIES

Policies are important because they:

- Policies clearly outline what actions are acceptable and unacceptable
- Procedures provide actionable steps when policies are violated
- Adhering to policies and procedures protects employees and organisations
- Not following policies exposes organisations to lawsuits

FEATURES / PRINCIPLES FOR GOOD GOVERNANCE OF IT

1. Responsibility - Individuals and groups within the organisation should understand and accept their responsibilities in respect of the supply of and the demand for IT. Those with responsibility for actions also must have the authority to perform those actions.
2. Strategy - The organisation's business strategy should takes into account the current and future capabilities of IT; the strategic plans for IT satisfy the current and ongoing needs of the organisation's business strategy.
3. Acquisition - IT acquisitions should be made for valid reasons and on the basis of appropriate and ongoing analysis, with clear and transparent decision making. There should be appropriate balance between benefits, opportunities, costs and risks, in both the short term and the long term.
4. Performance - IT is fit for purpose in supporting the organisation and in providing the services, the levels of service, and the service quality required to meet current and future business requirements.
5. Conformance - IT complies with all mandatory legislation and regulations. Policies and practices are clearly defined, implemented and enforced.
6. Human behaviour - IT policies, practices and decisions should demonstrate respect for human behaviour, including the current and evolving needs of all the "people in the process."

Applications of IT governance

- Help in decision making
- Ensures resource balancing,
- Allows risk management,
- Used in Law/regulation enforcement.

Policies are required in the following areas

Business Continuity –

- This is a statement of intent that seeks to minimise the impact that a disruptive incident could have on the Critical Business Functions of the University, including its teaching and learning, research and administration.
- The goal of a business continuity policy is to document what is needed keep an organisation running on ordinary business days as well as times of emergency. The policy can also be used to determine what went wrong so the problems can be addressed

Disaster Recovery

- A *disaster recovery plan/polict* (*DRP*) is a documented, structured approach that describes how an organization can quickly resume work after an unplanned incident.

Risk Management

- A risk management policy statement is a tool used by companies and organisations to identify and respond to risks in a way that minimizes their impact. The document outlines the risks that may face an organisations and steps to follow in case that risk occur.

User Policy.

- A document that outline what is required of a user when using an organisation's resource. It includes the responsibilities of an individual and the effect of not following the rules in the document.

https://searchdisasterrecovery.techtarget.com/definition/disaster-recovery-plan

IT CHANGEOVER

• Organisations have systems that operate in harmony with each other.
• After sometimes the systems cease to be efficient or don't do what they are supposed to do
• Sometimes the goals of the organisation changes and this calls for replacement of the old system e.g. when a new phone comes into the market most people want to acquire the new phone. Why

Need for Change Over

- Obsolete hardware making maintenance/repairs difficult or expensive
- Software will not run on new hardware
- Software is 'out of support',
- New versions of hardware or software have features that are desired/needed
- Existing hardware or software is insufficient for an expanding organisation
- Changes in an organisations role/operating methods make existing hardware or software unsuitable
- Improving security,
- Improving efficiency,
- Complying with new laws or regulations,
- Saving money

Changeover methods include:

- **Phased implementation:**
  - One part of the overall system that need changing is changed. If any problems arise due to changeover, they are dealt with thus not affecting the whole organisation.
  - Once the system has been successfully changed in one area, the other areas can follow suit, with any lessons learned during the initial changeover used to ensure the success of the changeover as a whole.
  - For example Implementation of the New curriculum in Kenya. It was started in the Lower classes, Class one and two, as the year goes by it is introduced in the next class until it covers all the classes.
- **Parallel running:**
  - Both the old and the new systems run side-by-side, using live data, so that project managers can compare the efficiency and reliability of the new system.
  - Once they are satisfied, the old system is taken offline and the new system becomes fully active and utilised across the organisation.
- **Direct changeover:**
  - In this method of implementation the old system stops being used and the new one becomes live.
  - This is the cheapest, quickest and easiest form of system changeover but is also the riskiest – if the system is broken or inefficient, the whole organisation suffers.
- **Pilot**
  - A part of the new system is implemented and observed. Any problems are dealt with.
  - If the system runs well, problems encountered are solved in the remaining system and the rest of the system is implemented in the whole organisation.

- **Pilot Changeover**
  - The new system is tried out at a test site before launching it company-wide.
  - For example, a bank may first test the system at one of its branches. This branch is referred to as the pilot, or beta, site for the program. If the system works in that branch then it is introduces in all the branches
  - Pilot changeover technique allows companies to run the new system next to their old but on a much smaller scale. This makes the pilot changeover method much more cost-effective. After the kinks are worked out of the system at the test site,

Resources

https://ki.se/sites/default/files/small_scale_oskar_kuus.pdf
www.wseas.us/e-library/conferences/2011/Jakarta/EACT/EACT-36.pdf
https://accountsiq.com/blog/accounting-system-changeover-heres-a-few-relevant-pointers/

Questions

1. Describe three advantages and three disadvantages of each method of changeover
2. Which method would you select for implementing a new system in a
    a. Hospital
    b. Company with several branches

14.2.3 SYSTEM MAINTENANCE.

- System Maintenance is the process of **monitoring**, **evaluating**, and **modifying** of existing information systems to make required or desirable improvements may be termed as

Types of maintenance

Corrective

- It deals with the repair of faults or defects found in day-today system functions. A defect can result due to errors in software design, logic and coding.
- Design errors occur when changes made to the software are incorrect, incomplete, wrongly communicated, or the change request is misunderstood.
- Logical errors result from invalid tests and conclusions, incorrect implementation of design specifications, faulty logic flow, or incomplete test of data.
- All these errors prevent the software from conforming to its agreed specifications. Note that the need for corrective maintenance is usually initiated by bug reports drawn by the users.
- In the event of a system failure due to an error, actions are taken to restore the operation of the software system.
- The approach in corrective maintenance is to locate the original specifications in order to determine what the system was originally designed to do. However, due to pressure from management, the maintenance team sometimes resorts to emergency fixes known as **patching**.

Adaptive

- This is the implementation of changes in a part of the system, which has been affected by a change that occurred in some other part of the system.
- It consists of adapting software to changes in the environment such as the hardware or the operating system. The term environment in this context refers to the conditions and the influences which act (from outside) on the system. For example, business rules, work patterns, and government policies have a significant impact on the software system.
- For instance, a government policy to use a single 'European currency' had a significant effect on the software system. An acceptance of this change required banks in various member countries to make significant changes in their software systems to accommodate this currency.

Perfective

- This is concerned with the change in the software that occurs while adding new functionalities in the software.
- It mainly deals with implementing new or changed user requirements.
- Perfective maintenance involves making functional enhancements to the system in addition to the activities to increase the system's performance even when the changes have not been suggested by faults.
- This includes enhancing both the function and efficiency of the code and changing the functionalities of the system as per the users' changing needs.
- Examples of perfective maintenance include modifying the payroll program to incorporate a new union settlement and adding a new report in the sales analysis system.
- Perfective maintenance accounts for 50%, that is, the largest of all the maintenance activities.

Preventive

- involves performing activities to prevent the occurrence of errors. It tends to reduce the software complexity thereby improving program understandability and increasing software maintainability.
- It comprises documentation updating, code optimisation, and code restructuring. Documentation updating involves modifying the documents affected by the changes in order to correspond to the present state of the system.
- Code optimisation involves modifying the programs for faster execution or efficient use of storage space. Code restructuring involves transforming the program structure for reducing the complexity in source code and making it easier to understand.

Planning for change

- When planning for change consider the following issues
- **Responsibilities**- Define the responsibilities of all individuals who are going to be concerned with the changes. Their roles or what is expected of them should be outlined
- **Documentation**- The changes that are being implemented should be well documented. Documentation help people who are making modification to the system understand how the system is working. Changes made to the system should be well documented.
- **Timing** – The timings when to implement the changes is important. Major changes may require the system to be shut down. Other changes may only require updating of a file. It is important the timings when to do the change is discussed and agreed upon.

Reasons for making changes

- Maintaining protection/security
- Meet level of service/uptime goals
- Reduce costs (prevention is usually cheaper than an emergency fix)
- Ensuring scheduled tasks such as backup continue to run smoothly
- Needed for warranty to be valid
- Legal requirements such as health and safety or data security laws

Resources

There is an example of a maintenance plan template here:
https://www2.gov.bc.ca/.../templates/nrs_maintenance_plan_template.docx

There is an overview of maintenance planning here: https://www.sebokwiki.org/wiki/System_Maintenance

14.2.4 ARCHIVING DATA

Definition

- The process of moving data that is no longer actively used to a separate storage device for long-term retention. Archive data consists of older data that remains important to the organisation or must be retained for future reference or regulatory compliance reasons.

- The process by which inactive information, in any format, is securely stored for long periods. Such information may – or may not – be used again in the future, but nonetheless should be stored until the end of its retention schedule.

- It should be emphasised that archived, inactive data can be made active again. Information should still be readily available if required at short notice.
- Archived data can be stored in a number of different formats, on a variety of devices.
- When businesses archive data, they tend to archive entire collections of files.
- Data that is archived might be needed again because of legal rulings or because it is vital information that is likely to be used again in the future.

Why achieve data?

- Meeting legal/regulatory/internal audit requirements,
- Easier access to older data,
- Easier analysis of historical/long term trends,
- Making current database/data storage more manageable/more responsive/cheaper,
- More secure,
- Less likely to lose/accidentally delete data,
- Reduce time/storage/other requirements of normal backups,
- Protection of intellectual property,
- For the purposes of disaster recovery

Data archiving vs. backup

Data archives are not to be confused with data backups, which are copies of data. Although both are considered secondary storage and use a lower-performance, higher-capacity storage medium than primary storage, they serve different purposes. Archives fill a data retention purpose, whereas backups are used for data protection and disaster recovery.

Data archives can be thought of as a data repository for infrequently accessed, but still readily available data. Backups, on the other hand, are part of a data recovery mechanism that can be used to restore data in the event it is corrupted or destroyed. Backup data often consists of important information that must be restored quickly when lost or deleted.

Disadvantages of Archiving

- Restore time
- Possible obsolescence of storage media/hardware/data formats
- Ongoing maintenance costs,

Resources

http://www.nationalarchives.gov.uk/
https://uk-air.defra.gov.uk/data/
https://archive.org/web/

14.2.5 DISASTER RECOVERY PLANS.

- A disaster recovery plan (DRP) is a documented, structured approach with instructions for responding to unplanned incidents.
- This step-by-step plan consists of the precautions to minimise the effects of a disaster so the organisation can continue to operate or quickly resume mission-critical functions.
- Typically, disaster recovery planning involves an analysis of business processes and continuity needs.
- Before generating a detailed plan, an organisation often performs a business impact analysis (BIA) and risk analysis (RA), and it establishes the recovery time objective (RTO) and recovery point objective (RPO).
- A DR plan checklist includes the following steps
    - Establishing the scope of the activity;
    - Gathering relevant network infrastructure documents;
    - Identifying the most serious threats and vulnerabilities, and the most critical assets;
    - Reviewing the history of unplanned incidents and outages, and how they were handled;
    - Identifying the current DR strategies;
    - Identifying the emergency response team;
    - Having management review and approve the disaster recovery plan;
    - Testing the plan;
    - Updating the plan; and
    - Implementing a DR plan audit

Reasons for Disaster Recovery Planning

- Downtime is expensive and can cause a business to fail
- Customers expect 24/7 access
- A disaster is almost a certainty (hardware failure, power supply problems, natural events),
- Legal/regulatory requirements
- Data loss can kill a company (over 90% of organisations that suffer a major data loss close within two years),
- People make mistakes/human error is almost unavoidable
- External threats such as ransomware

Types of Plan

- **Preventative** –
    - Measures to stop the disaster affecting the system e.g. surge protection, anti-flood measures, fire suppression systems, inspection and maintenance,
- **Detective** –
    - Looking for and mitigating new threats e.g. keeping antivirus updated, training users, using network monitoring tools
- **Corrective** –
    - Restoring the system after a disaster has happened

Features of Disaster Recovery Plans

- business continuity plan
- Responsibilities of each individual
- Documentation,
- Redundancy,
- Prioritisation,
- Communication
- Testing.

Types of disasters

A disaster recovery plan protects an organisation from both human-made and natural disasters. There is not one specific way to recover from all kinds of disasters, so a plan should tackle a range

of possibilities. A natural disaster may seem unlikely, but if it can happen in the organisation's location, the DR plan should address it.

Potential disasters should include:

- Application failure
- VM failure
- Natural disasters such as earthquakes
- Hardware failures
- Fires
- Thefts of both hardwares

Testing your disaster recovery plan

- DR plans are substantiated through testing, which identifies deficiencies and provides opportunities to fix problems before a disaster occurs. Testing can offer proof that the plan is effective and hits RPOs and RTOs. Since IT systems and technologies are constantly changing, DR testing also helps ensure a disaster recovery plan is up to date.
- Reasons given for not testing DR plans include budget restrictions, resource constraints or a lack of management approval. Disaster recovery testing takes time, resources and planning. It can also be a risk if the test involves using live data.
- DR testing can be simple to complex. In a plan review, a detailed discussion of the disaster recovery plan looks for missing elements and inconsistencies. In a table top test, participants walk through plan activities step by step to demonstrate whether disaster recovery team members know their duties in an emergency. A simulation test uses resources such as recovery sites and backup systems in what is essentially a full-scale test without an actual failover.

TOPIC 15: SYSTEMS DEVELOPMENT

15.1.1

What is a project?

- A project is a temporary endeavour undertaken to create a unique product, service, or result.
- The temporary nature of projects indicates that a project has a definite beginning and end. The end is reached when the project's objectives have been achieved or when the project is terminated because its objectives will not or cannot be met, or when the need for the project no longer exists. A project may also be terminated if the client (customer, sponsor, or champion) wishes to terminate the project.
- Every project creates a unique product, service, or result. The outcome of the project may be tangible or intangible.

- Examples of projects include:
    - Developing a new product, service, or result;
    - Effecting a change in the structure, processes, staffing, or style of an organisation;
    - Developing or acquiring a new or modified information system (hardware or software);
    - Conducting a research effort whose outcome will be aptly recorded;
    - Constructing a building, industrial plant, or infrastructure; or
    - Implementing, improving, or enhancing existing business processes and procedures.

What is Project Management

- Project management is the application of knowledge, skills, tools, and techniques to project activities to meet the project requirements.
- Project management is accomplished through the appropriate application and integration of project management processes

Features of Project Management

- **Project Integration Management** - includes the processes and activities to identify, define, combine, unify, and coordinate the various processes and project management activities
- **Planning and Scheduling** -
- **Collaboration** -
- **Documentation** -
- **Reporting** -
- **Resource Management** -
- **Budget Management** -
- **Risk Management** -

Why carry out Project Management?

- The fundamentals of managing a project from start to finish require a team of individuals with different talents and skills. Those people are responsible for planning and executing the project objectives and that takes more than just labour and materials.
- Each project follows a Project Life Cycle. A sound project plan can mean the difference between success or failure. Each project follows a Project Life Cycle.

1. Defining The Scope of the Project – Project Scope Management includes the processes required to ensure that the project includes all the work required, and only the work required, to complete the project successfully. Managing the project scope is primarily concerned with defining and controlling what is and is not included in the project.

2. Defines a plan and organises chaos – projects are naturally chaotic. The primary business function of project management is organizing and planning projects to tame this chaos. A clear path mapped out from start to finish ensures the outcome meets the goals of your project.

2. Establishes a schedule and plan – Without a schedule, a project has a higher probability of delays and cost overruns. A sound schedule is key to a successful project.

3. Enforces and encourages teamwork – A project brings people together to share ideas and provide inspiration. Collaboration is the cornerstone to effective project planning and management.

4. Maximises resources – Resources, whether financial or human, are expensive. By enforcing project management disciplines such as project tracking and risk management, all resources are used efficiently and economically.

5. Manages Integration – Projects don't happen in a vacuum. They need to be integrated with business processes, systems and organisations.
You can't build a sales system that doesn't integrate with your sales process and sales organisation. It wouldn't add much value. Integration is often key to project value.
Project management identifies and manages integration.

6. Controls cost – some projects can cost a significant amount of money so on budget performance is essential. Using project management strategies greatly reduces the risk of budget overruns.

7. Manages change – projects always happen in an environment in which nothing is constant except change. Managing change is a complex and daunting task. It is not optional. Project management manages change.

8. Managing quality – Quality is the value of what you produce. Project management identifies, manages and controls quality. This results in a high quality product or service and a happy client.

9. Retain and use knowledge – projects generate knowledge or at least they should. Knowledge represents a significant asset for most businesses. Left unmanaged knowledge tends to quickly fade. Project management ensures that knowledge is captured and managed.

10. Learning from failure – projects do fail. When they do, it is important to learn from the process. Project management ensures that lessons are learned from project success and failure.

- identifying key issues and assumptions,
- tracking progress,
- formalising/standardising methods/policies/strategies,
- establishing leadership/chain of command/responsibilities,
- optimising use of resources,
- cost control,
- integrating the project with the wider business/organisational environment,
- retaining skills/knowledge between projects,
- learning from mistakes/failures

Characteristics of Successful Projects;

- **Fulfils Criteria** – does what it was supposed to accomplish
- **Within Budget** – used the resources allocated
- **On Time**- It is delivered on time according to the plan.
- **Satisfying Stakeholders** – stakeholders are the people who have an interest or are going to be affected by the project in one way or another. They include the customers, consumers, workers and donors

https://thedigitalprojectmanager.com/why-is-project-management-important/
https://2020projectmanagement.com/resources/general-interest-and-miscellaneous/10-reasons-why-project-management-matters

15.1.13 SMART Objectives

- When carrying out a project, it's important to develop objectives.
- Objectives are the measures that will be used to determine whether the projective meet its goals.
- Objectives used to evaluate a project must be **SMART**.
- SMART objectives are **Specific, Measurable, Achievable, Relevant,** and **Time-related**.
- **Specific**
    - It answers the questions **"what is to be done?"** "**how will you know it is done?"** and **describes the results (end product) of the work to be done**.
    - A person must be conversant with the field of study and must know what end product is required.
    - To ensure that an objective is specific is to make sure that the way it is described is observable.
    - Observable means that somebody can see or hear (physically observe) someone doing something.
- **Measurable w/Measurement**
    - This answers the question "**how will you know it meets expectations**?" and defines the objective using assessable terms (quantity, quality, frequency, costs, deadlines, etc.). It refers to the extent to which something can be evaluated against some standard. An objective with a quantity measurements uses terms of amount, percentages, etc...
- **Achievable**
    - Achievable answers the questions "**can the person do it**?" "Can the measurable objective be achieved by the person?" "Does he/she have the experience, knowledge or capability of fulfilling the expectation?"
    - It also answers the question "Can it be done giving the time frame, opportunity and resources?"
- **Relevant**
    - Relevant answers the questions, "**should it be done**?", "**why**?" and "what will be the impact?"
    - Is the objective aligned with the implementation plan and the organisation's strategic plan?
- **Time-oriented**
    - Time-oriented answers the question, "**when will it be done**?"
    - It refers to the fact that an objective has end points and check points built into it.
    - Everything has a beginning and an end. Time-oriented states the duration its going to take to complete the task.

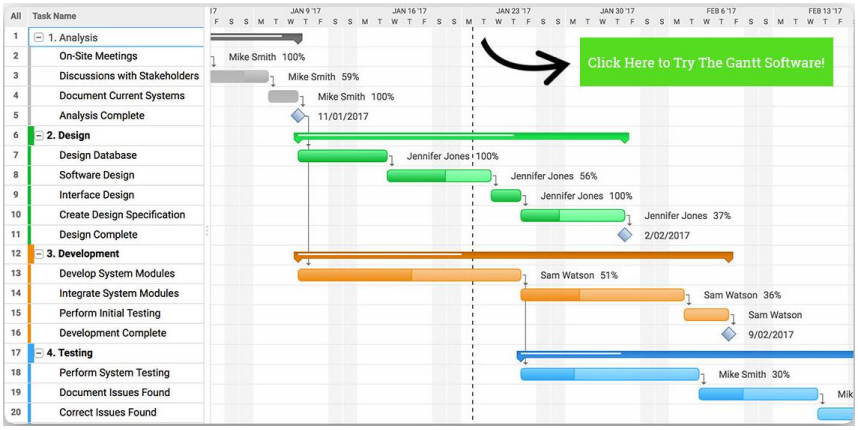
https://www.businessballs.com/project-management/project-management-skills-and-techniques/
https://www.toolshero.com/time-management/smart-goals/

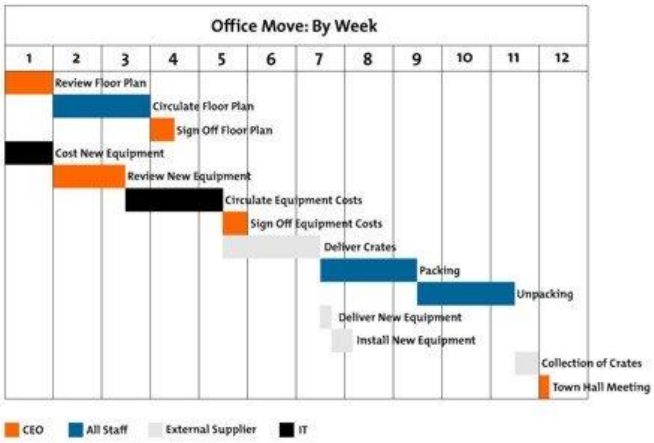
Project Management Tools

- A project consists of several activities that are either being carried out concurrently or one after another. To complete a project successfully, you must control these activities, and ensure that they are completed on schedule.
- Missing a deadline or finishing a task out of sequence could have a knock-on effects on the rest of the project. It could deliver late a result, and cost a lot more.
- Its therefore helpful to be able to see everything that needs to be done, and know, at a glance, when each activity needs to be completed.
- Project management tools assist in doing that.
- Project Management tools include

Gant Charts

- This is a visual tool that allow a user to see tasks scheduled over time.
- It outlines all of the tasks involved in a project, and their order, shown against a timescale. This gives a person an instant overview of a project, its associated tasks, and when these need to be finished
- Gantt Charts list the tasks to be performed on the vertical axis, and time intervals on the horizontal axis.
- The width of the horizontal bars in the graph shows the duration of each activity.
- The also indicate the person who is responsible for that task
- Gantt charts illustrate the start and finish dates of the activities





Example of a Gantt Chart

- On a Gantt chart, you can easily see:
  - The start date of the project
  - What the project tasks are
  - Who is working on each task
  - When tasks start and finish
  - How long each task will take
  - How tasks group together, overlap and link with each other

- The finish date of the project.

How to Draw a Gantt Chart

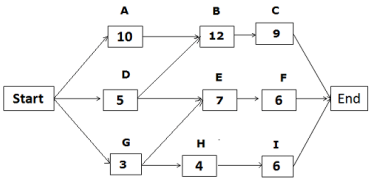
1. Identify Essential Tasks of any project
2. Identify Task Relationships that exist between these tasks.
    - In Gantt charts, there are three main relationships between sequential tasks:
    - **Finish to Start (FS)** – FS tasks cannot start before a previous (and related) task is finished. However, they can start later. Eg. You cannot fix window frames before completing the house structure or you cannot start roofing until the walls or beams have been erected
    - **Start to Start (SS)** – SS tasks cannot start until a preceding task starts. However, they can start later.
    - **Finish to Finish (FF)** – FF tasks cannot end before a preceding task ends. However, they can end later. E.g – Plastering of the house cannot finish before the door frames have been fixed
    - **Start to Finish (SF),** is very rare
3. Draw your charts by hand or use specialist software

Question:
Draw a Gantt chart for a building a house

https://www.mindtools.com/pages/article/newPPM_03.htm

Critical Path Analysis

- Critical path analysis (CPA) is a project management technique that requires mapping out every key task that is necessary to complete a project.
- It requires
    - identifying the tasks
    - the amount of time necessary to finish each activity
    - the dependencies of each activity on any others.
- Critical Path Analysis formally identifies tasks, which must be completed on time for the whole project to be completed on time.
- It also identifies which tasks can be delayed if resources need to be reallocated to catch up on missed or overrunning tasks.
- It is used to set a realistic deadline for a project and to track its progress along the way



- In the above network diagram, the path "Start-A-B-C-End" is the critical path as it has the longest duration.
- All activities on a critical path are known as critical activities and any delay in the completion of these activities will delay the project by the same amount of time.

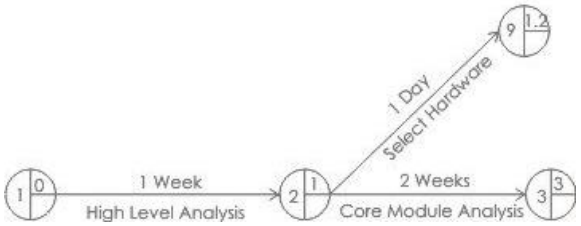
How to Draw a Critical Path Diagram

1. **List All Activities in the Plan**
   For each activity, show the **earliest start date**, **estimated length of time it will take**, and **whether it is parallel or sequential**. If tasks are sequential, show which stage they depend on.
2. **Plot the Activities as a Circle and Arrow Diagram**
   Critical Path Analyses are presented using circle and arrow diagrams.
   In these, circles show events within the project, such as the start and finish of tasks. The number shown in the left-hand half of the circle allows you to identify each one easily. Circles are sometimes known as nodes.

An arrow running between two event circles shows the activity needed to complete that task. A description of the task is written underneath the arrow. The length of the task is shown above it. By convention, all arrows run left to right. Arrows are also sometimes called arcs.

Example Critical Path Diagram: Step 1

Where one activity cannot start until another has been completed, we start the arrow for the dependent activity at the completion event circle of the previous activity.



Example Critical Path Diagram: Step 2

https://www.mindtools.com/pages/article/critical-path-analysis.htm

PERT (Program Evaluation and Review Technique)

PERT is a variation on Critical Path Analysis that takes a slightly more sceptical view of time estimates made for each project stage.
To use it, estimate the shortest possible time each activity will take, the most likely length of time, and the longest time that might be taken if the activity takes longer than expected.

Use the formula below to calculate the time to use for each project stage:

$$\frac{\text{Shortest time} + 4 \times \text{likely time} + \text{longest time}}{6}$$

This helps to bias time estimates away from the unrealistically short time-scales normally assumed.
Key Points

Critical Path Analysis is an effective and powerful method of assessing:

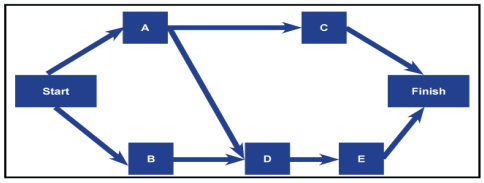
- What tasks must be carried out.
- Where parallel activity can be performed.
- The shortest time in which you can complete a project.
- Resources needed to execute a project.
- The sequence of activities, scheduling and timings involved.
- Task priorities.
- The most efficient way of shortening time on urgent projects.

An effective Critical Path Analysis can make the difference between success and failure on complex projects. It can be very useful for assessing the importance of problems faced during the implementation of the plan.

PERT is a variant of Critical Path Analysis that takes a more skeptical view of the time needed to complete each project stage.

Precedence Diagramming Method (PDM)

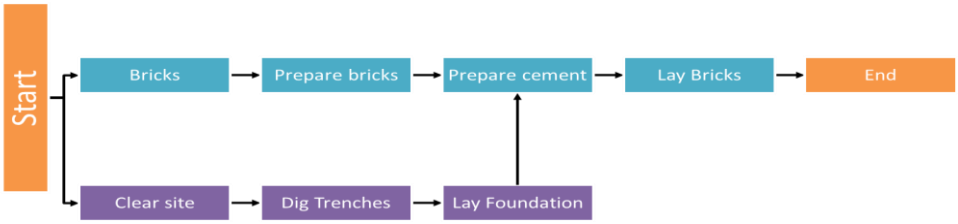
- Precedence Diagramming Method (PDM) is a strategy for developing a project schedule network diagram that utilizes nodes to represent activities and associates them with projectiles that illustrate the dependencies.
- This method is likewise called the activity-on-node (AON).



- The most significant advantage of using the Precedence Diagramming Method is that it quickly allows the project team to understand all the scheduled activities and its affiliates with each other.

Creating Precedence Diagramming Method (PDM)

As a project manager, ensure that your emphasis is on project sequence. As you put effort on the activities, you may come across additional tasks that have to be performed, and henceforth you can incorporate the equivalent in the precedence diagram and to the work breakdown structure. At last draw bolts between the assignments that connect up, and ensure that there aren't any tasks sitting isolated.



A Precedence Diagramming Method (PDM), which is sometimes also known as the Activity on Node (AON) Diagramming Method, is a graphical representation technique, which shows the inter-dependencies among various project activities.

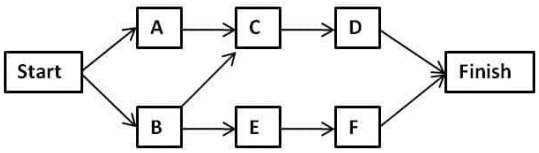
This diagramming method is utilized to draw the project schedule network diagrams—for example the critical path network diagram and the critical chain network diagram.

You may have also heard about another less commonly used technique in diagramming methods—the Activity on Arrow (AOA) diagramming method, which is a special case of the Precedence Diagramming Method.

In AOA, all dependencies are Finish to Start, and the duration is shown on arrows. That is why this diagramming method is known as the Activity on Arrow (AOA) diagram. PERT is an example of AOA diagram.

Activity on Arrow (AOA) diagram emphasizes milestones (events), and the AON diagram emphasizes tasks.

The main benefit of the Precedence Diagramming Method (PDM) is that it shows the activity dependencies, and it can be an important communication tool for stakeholders.

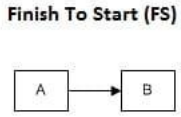


Precedence Diagram Method (PDM)

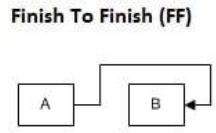
- **The Precedence Diagramming Method (PDM)** consists of rectangles, known as nodes, and the project activities are shown in these boxes.
- These rectangular boxes are connected with arrows to show the dependencies; therefore, these diagrams are also known as the **Activity on Node (AON)** diagrams.
- The Precedence Diagramming Method uses four types of dependencies. Those dependencies are as follows:
  1. Finish to Start (FS)
  2. Finish to Finish (FF)
  3. Start to Start (SS)
  4. Start to Finish (SF)

  - **Finish to Start (FS)** In this type of dependency, the second activity cannot be started until the first activity completes. This type of dependency is the most commonly used dependency in the diagramming techniques.

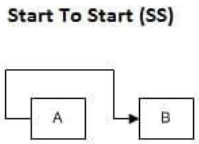
- For example, to paint a wall you first need to build a wall. In this case, the first activity is building the wall and second activity will be painting. You cannot start painting the wall unless the wall is ready.



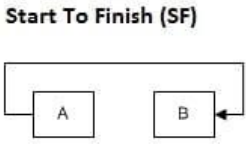
- **Finish to Finish (FF)**Here, the second activity cannot be finished until the first activity finishes. In other words, both activities should finish simultaneously.
- For example, let us say that you are coding a program for a client, and the client is providing you with the characteristics of the program. In this case, you cannot finish coding for your program until the client gives you his complete requirements. Here, both activities should finish simultaneously.



- **Start to Start (SS)-** Here, the second activity cannot be started until the first activity starts. Both activities should start simultaneously.
- Suppose you have to apply a primary coating on a wall. To apply the coating, you also need to clean the wall. Therefore, one team will start cleaning the wall and second team will paint it. Both activities can be started at the same time.



- **Start to Finish (SF) -** In this type of dependency, the second activity cannot be finished until the first activity starts.
- For example, let us say you have to move into a new home, and your old home has to be demolished. In this case, you cannot move to your new home until it is ready. Hence, the second activity (construction of new home) must be finished before the first activity starts (you start moving into a new home); i.e. if you are moving into your new home, you cannot start vacating your old home until the new house is completely ready.



Draw a PDM network for a project with the following activities. Show all steps including removing redundant relations; and sequence steps.
– Activity B depends on A;
– Activity G follows E, F & D;
– Activity E depends on B and A;
– Activity F can start when D & B are completed;
– Activity C is followed by F and follows A;
– Activity D is dependent upon A and B.
https://www.invensislearning.com/resources/pmp/precedence-diagramming-method

Precedence tables

- A precedence table is a tool for scheduling activities in planning a project.
- It is based on the logic of what activities must follow other activities and what activities can run at the same time.
- Any project, whether it's cooking a meal or building a house, can be broken down into a set of separate activities.

- Some activities cannot be started before others have been finished.
- For example, when building a house you cannot put in the windows before you have built the walls.

Tea making

Making a cup of tea involves the following activities:

Activity	
A	Boil the kettle
B	Put a tea bag in a mug
C	Brew the tea
D	Add milk

But there are constraints as some activities depend on each other:

- Brewing the tea cannot happen before you have boiled the kettle
- Brewing the tea cannot happen you have put a tea bag in a mug
- Adding the milk cannot happen before you have brewed the tea

A precedence table helps you organise the activities and shows where there are constraints caused by dependencies. A **dependency** is where carrying an activity **depends** on another activity having been carried out.
Notice that a dependency only refers to the activity immediately before the activity in question.

Activity	**Depends on**
A	
B	
C	A, B
D	C

From this table you can see that either A or B could be started first as they do not depend on any other activity.

Building an extension

Question

Make a precedence table showing the constraints for building an extension.

Activity	
A	Get planning permission
B	Build the foundations
C	Build the walls
D	Put in windows and doors
E	Build the roof
F	Put in electrics
G	Put in plumbing
H	Decorating

The constraints are:

- B depends on A
- C depends on B
- D and E depend on C
- F and G depend on E
- H depends on F and G

Activity	**Depends on**
A	
B	A
C	B
D	C

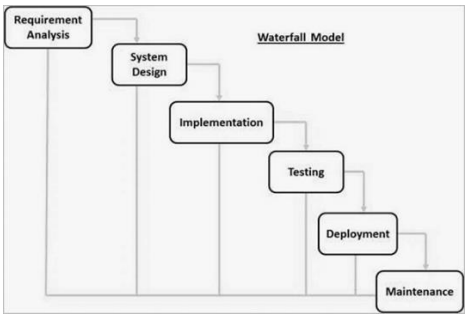
Activity	Depends on
E	C
F	E
G	E
H	F and G

Waterfall

- This is a method of developing a system. I was used for software development
- It is also referred to as a **linear-sequential life cycle model**. It is very simple to understand and use.
- In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases.
- The Waterfall model is the earliest **SDLC** approach that was used for software development.
- The waterfall Model illustrates the software development process in a linear sequential flow. This means that any phase in the development process begins only if the previous phase is complete. In this waterfall model, the phases do not overlap.

Waterfall Model - Design

- Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project.
- In "The Waterfall" approach, the whole process of software development is divided into separate phases.
- In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially.
- The following illustration is a representation of the different phases of the Waterfall Model.



The sequential phases in Waterfall model are −

- **Requirement Gathering and analysis** − All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** − The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** − With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing** − All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system** − Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance** − There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

- All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases.
- The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model, phases do not overlap.

Why use water fall model?

- Requirements are very well documented, clear and fixed.
- Product definition is stable.
- Technology is understood and is not dynamic.
- There are no ambiguous requirements.
- Ample resources with required expertise are available to support the product.
- The project is short.

Waterfall Model - Advantages

- Simple and easy to understand and use
- Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
- Phases are processed and completed one at a time.
- Works well for smaller projects where requirements are very well understood.
- Clearly defined stages.
- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented.

Waterfall Model - Disadvantages

- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
- It is difficult to measure progress within stages.
- Cannot accommodate changing requirements.
- Adjusting scope during the life cycle can end a project.
- Integration is done as a "big-bang. at the very end, which doesn't allow identifying any technological or business bottleneck or challenges early.

Agile Process and Method

What is Agile Methodology?

- This is a practice that promotes continuous iteration of development and testing throughout the software development lifecycle of the project.
- Both development and testing activities are concurrent

- The agile software development emphasizes on four core values.
    - Individual and team interactions over processes and tools
    - Working software over comprehensive documentation
    - Customer collaboration over contract negotiation
    - Responding to change over following a plan

A sprint is a period of time allocated for a particular phase of a project. Sprints are considered to be complete when the time period expires. There may be disagreements among the members of the team as to whether or not the development is satisfactory; however, there will be no more work on that particular phase of the project. The remaining phases of the project will continue to develop within their respective time frames.

The General Principles of the Agile Method

- Satisfy the client and continually develop software.
- Changing requirements are embraced for the client's competitive advantage.
- Concentrate on delivering working software frequently. Delivery preference will be placed on the shortest possible time span.
- Developers and business people must work together throughout the entire project.
- Projects must be based on people who are motivated. Give them the proper environment and the support that they need. They should be trusted to get their jobs done.
- Face-to-face communication is the best way to transfer information to and from a team.
- Working software is the primary measurement of progress.
- Agile processes will promote development that is sustainable. Sponsors, developers, and users should be able to maintain an indefinite, constant pace.
- Constant attention to technical excellence and good design will enhance agility.
- Simplicity is considered to be the art of maximizing the work that is not done, and it is essential.
- Self-organized teams usually create the best designs.
- At regular intervals, the team will reflect on how to become more effective, and they will tune and adjust their behaviour accordingly.

Benefits of Using the Agile Method

The Agile Method grew out of the experience with the real-life projects of leading software professionals from the past. Because of this, the challenges and limitations of traditional development have been discarded. Subsequently, the Agile Method has been accepted by the industry as a better solution to project development. Nearly every software developer has used the Agile Method in some form.

This method offers a light framework for assisting teams. It helps them function and maintain focus on rapid delivery. This focus assists capable organisations in reducing the overall risks associated with software development.

The Agile Method ensures that value is optimized throughout the development process. The use of iterative planning and feedback results in teams that can continuously align a delivered product that reflects the desired needs of a client. It easily adapts to changing requirements throughout the process by measuring and evaluating the status of a project. The measuring and evaluating allows accurate and early visibility into the progress of each project.

It could be stated that the Agile Method helps companies build the right product. Instead of trying to market software before it is written, the Agile Method empowers teams to optimize the release during its development. This allows the product to be as competitive as possible within the marketplace. It preserves the relevance of the critical market, and it ensures that a team's work doesn't wind up collecting dust on a shelf. This is why the Agile Method is an attractive developmental option for stakeholders and developers alike.

There are many critics of the Agile Method; however, this method produces results that clients can take to the bank. Although a project may not turn out exactly as the client envisions, it will be delivered within the time that it needs to be produced. Throughout the process, the client and the team are changing the requirements in order to produce the quality needed by the client. Clients are happy with the results, and the team satisfies the client's needs. The ongoing change can sometimes give both the client and the team more than they had originally envisioned for the product. The Agile Method really is a winning solution for everyone involved in software development.

Criticism of Agile Development

- It is developer-centric rather than user-centric.
- Agile focuses on processes for getting requirements and developing code and does not focus on product design.
- Agile methodologies can also be inefficient in large organisations and certain types of projects.

Difference Between Agile and Traditional (Waterfall or Spiral) Development

Fundamental Assumptions

Traditional: Systems are fully specifiable, predictable, and can be built through meticulous and extensive planning.

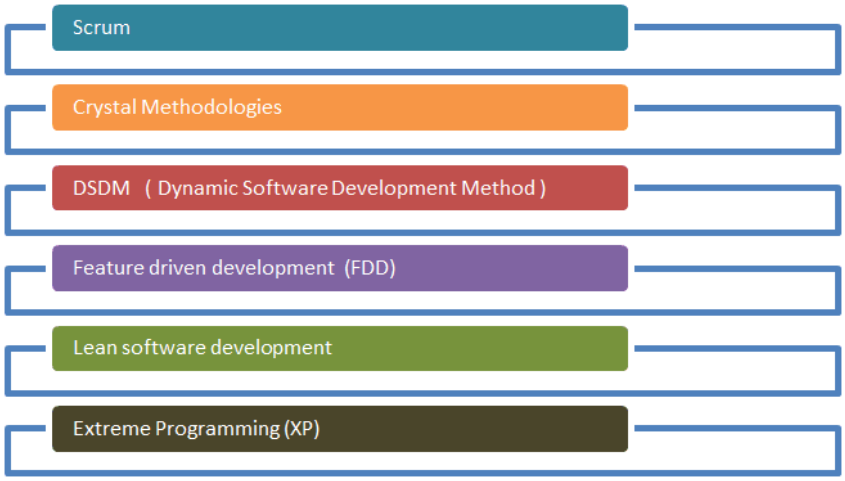
Agile: High-quality, adaptive software can be developed by small teams using the principles of continuous design improvement and testing based on rapid feedback and change.

Control
Traditional: Process-centric
Agile: People-centric
Management Style
Traditional: Command-and-control
Agile: Leadership-and-collaboration
Knowledge Management
Traditional: Explicit
Agile: Tacit
Role Assignment
Traditional: Individual—favors specialization
Agile: Self-organizing teams—encourages role interchangeability
Communication
Traditional: Formal
Agile: Informal
Customer's Role
Traditional: Important
Agile: Critical
Project Cycle
Traditional: Guided by tasks or activities
Agile: Guided by product features
Development Model
Traditional: Life cycle model (Waterfall, Spiral, or some variation)
Agile: The evolutionary-delivery model
Desired Organisational Form/Structure
Traditional: Mechanistic
Agile: Organic
Technology
Traditional: No restriction
Agile: Favors object-oriented technology

Agile Vs Waterfall Method

Agile Model	Waterfall Model
Agile method proposes incremental and iterative approach to software design	Development of the software flows sequentially from start point to end point.
The **agile process** is broken into individual models that designers work on	The design process is not broken into an individual models
The customer has early and frequent opportunities to look at the product and make decision and changes to the project	The customer can only see the product at the end of the project
Agile model is considered unstructured compared to the waterfall model	Waterfall model are more secure because they are so plan oriented
Small projects can be implemented very quickly. For large projects, it is difficult to estimate the development time.	All sorts of project can be estimated and completed.
Error can be fixed in the middle of the project.	Only at the end, the whole product is tested. If the requirement error is found or any changes have to be made, the project has to start from the beginning
Development process is iterative, and the project is executed in short (2-4) weeks iterations. Planning is very less.	The development process is phased, and the phase is much bigger than iteration. Every phase ends with the detailed description of the next phase.
Documentation attends less priority than software development	Documentation is a top priority and can even use for training staff and upgrade the software with another team
Every iteration has its own testing phase. It allows implementing regression testing every time new functions or logic are released.	Only after the development phase, the testing phase is executed because separate parts are not fully functional.
In agile testing when an iteration end, shippable features of the product is delivered to the customer. New features are usable right after shipment. It is useful when you have good contact with customers.	All features developed are delivered at once after the long implementation phase.
Testers and developers work together	Testers work separately from developers
At the end of every sprint, user acceptance is performed	User acceptance is **performed** at the end of the project.
It requires close communication with developers and together analyse requirements and planning	Developer does not involve in requirement and planning process. Usually, time delays between tests and coding

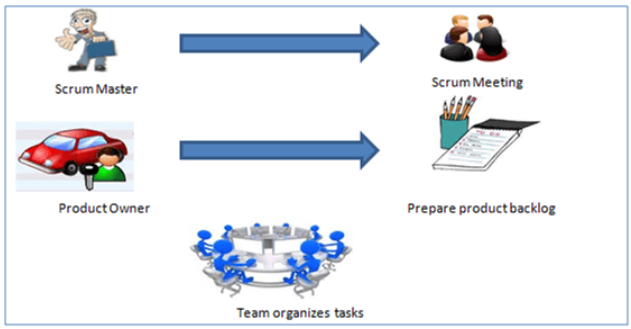
Agile Methodology



There are various **methods** present in agile testing, and those are listed below:

Scrum

- SCRUM is an agile development method which concentrates specifically on how to manage tasks within a team-based development environment. Basically, Scrum is derived from activity that occurs during a rugby match. Scrum believes in empowering the development team and advocates working in small teams (say- 7 to 9 members). It consists of three roles, and their responsibilities are explained as follows:



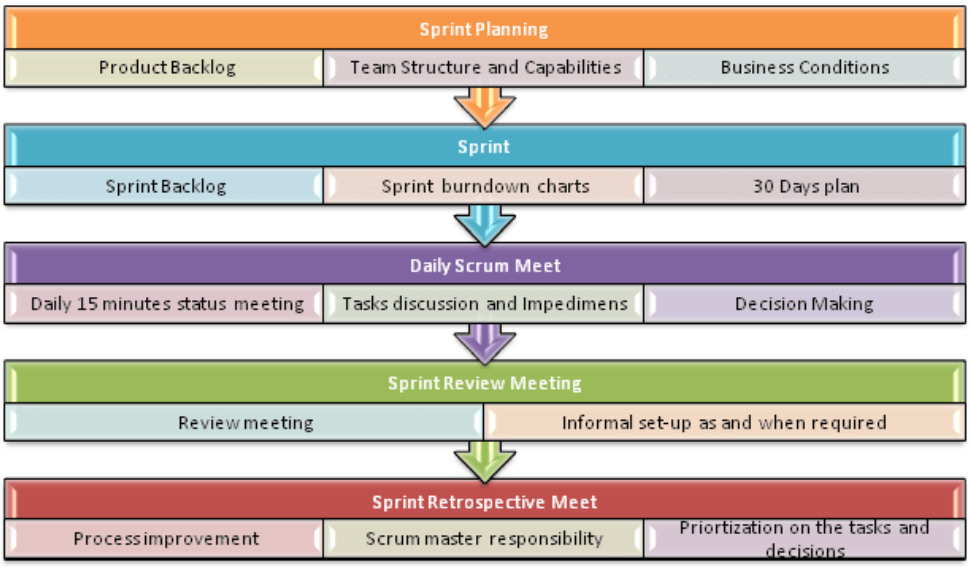
  - Scrum Master
    - Master is responsible for setting up the team, sprint meeting and removes obstacles to progress
  - Product owner
    - The Product Owner creates product backlog, prioritizes the backlog and is responsible for the delivery of the functionality at each iteration
  - Scrum Team
    - Team manages its own work and organizes the work to complete the sprint or cycle

Product Backlog

This is a repository where requirements are tracked with details on the no of requirements(user stories) to be completed for each release. It should be maintained and prioritized by Product Owner, and it should be distributed to the scrum team. Team can also request for a new requirement addition or modification or deletion

Scrum Practices

Practices are described in detailed:



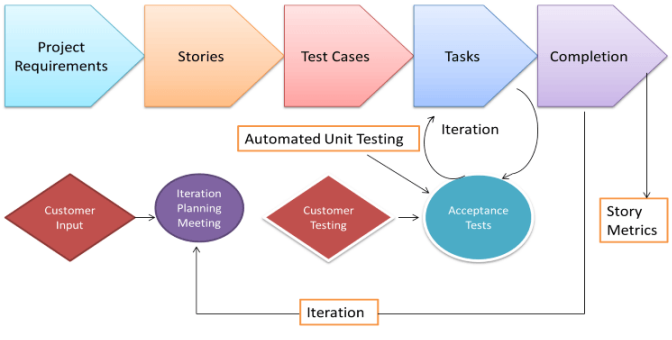
Process flow of Scrum Methodologies:

Process flow of scrum testing is as follows:

- Each iteration of a scrum is known as Sprint
- Product backlog is a list where all details are entered to get the end-product
- During each Sprint, top user stories of Product backlog are selected and turned into Sprint backlog
- Team works on the defined sprint backlog
- Team checks for the daily work
- At the end of the sprint, team delivers product functionality

eXtreme Programming (XP)

Extreme Programming technique is very helpful when there is constantly changing demands or requirements from the customers or when they are not sure about the functionality of the system. It advocates frequent "releases" of the product in short development cycles, which inherently improves the productivity of the system and also introduces a checkpoint where any customer requirements can be easily implemented. The XP develops software keeping customer in the target.



Extreme Programming (XP)

Business requirements are gathered in terms of stories. All those stories are stored in a place called the parking lot.

In this type of methodology, releases are based on the shorter cycles called Iterations with span of 14 days time period. Each iteration includes phases like coding, unit testing and system testing where at each phase some minor or major functionality will be built in the application.

Phases of eXtreme programming:

There are 6 phases available in Agile XP method, and those are explained as follows:

Planning

- Identification of stakeholders and sponsors
- Infrastructure Requirements
- Security related information and gathering
- Service Level Agreements and its conditions

Analysis

- Capturing of Stories in Parking lot
- Prioritize stories in Parking lot

- Scrubbing of stories for estimation
- Define Iteration SPAN(Time)
- Resource planning for both Development and QA teams

Design

- Break down of tasks
- Test Scenario preparation for each task
- Regression Automation Framework

Execution

- Coding
- Unit Testing
- Execution of Manual test scenarios
- Defect Report generation
- Conversion of Manual to Automation regression test cases
- Mid Iteration review
- End of Iteration review

Wrapping

- Small Releases
- Regression Testing
- Demos and reviews
- Develop new stories based on the need
- Process Improvements based on end of iteration review comments

Closure

- Pilot Launch
- Training
- Production Launch
- SLA Guarantee assurance
- Review SOA strategy
- Production Support

There are two storyboards available to track the work on a daily basis, and those are listed below for reference.

- Story Cardboard
    - This is a traditional way of collecting all the stories in a board in the form of stick notes to track daily XP activities. As this manual activity involves more effort and time, it is better to switch to an online form.
- Online Storyboard
    - Online tool Storyboard can be used to store the stories. **Several teams can use it** for different purposes.

Crystal Methodologies

Crystal Methodology is based on three concepts

1. **Chartering:** Various activities involved in this phase are creating a development team, performing a preliminary feasibility analysis, developing an initial plan and fine-tuning the development methodology
2. **Cyclic delivery:** The main development phase consists of two or more delivery cycles, during which the
    1. Team updates and refines the release plan
    2. Implements a subset of the requirements through one or more program test integrate iterations
    3. Integrated product is delivered to real users
    4. Review of the project plan and adopted development methodology
3. **Wrap Up:** The activities performed in this phase are deployment into the user environment, post- deployment reviews and reflections are performed.

Dynamic Software Development Method (DSDM)

DSDM is a Rapid Application Development (RAD) approach to software development and provides an agile project delivery framework. The important aspect of DSDM is that the users are required to be involved actively, and the teams are given the power to make decisions. Frequent delivery of product becomes the active focus with DSDM. The techniques used in DSDM are

1. Time Boxing
2. MoSCoW Rules
3. Prototyping

The DSDM project consists of 7 phases

1. Pre-project
2. Feasibility Study
3. Business Study
4. Functional Model Iteration
5. Design and build Iteration
6. Implementation
7. Post-project

Feature Driven Development (FDD)

This method is focused around "designing & building" features. Unlike other agile methods, FDD describes very specific and short phases of work that has to be accomplished separately per feature. It includes domain walkthrough, design inspection, promote to build, code inspection and design. FDD develops product keeping following things in the target

1. Domain object Modeling
2. Development by feature
3. Component/ Class Ownership
4. Feature Teams
5. Inspections
6. Configuration Management
7. Regular Builds
8. Visibility of progress and results

Lean Software Development

Lean software development method is based on the principle "Just in time production". It aims at increasing speed of software development and decreasing cost. Lean development can be summarized in seven steps.

1. Eliminating Waste
2. Amplifying learning
3. Defer commitment (deciding as late as possible)
4. Early delivery
5. Empowering the team
6. Building Integrity
7. Optimize the whole

Kanban

Kanban originally emerged from Japanese word that means, a card containing all the information needed to be done on the product at each stage along its path to completion. This framework or method is quite adopted in software testing method especially in agile testing.

Scrum Vs Kanban

Scrum	Kanban
In scrum technique, test must be broken down so that they can be completed within one sprint	No particular item size is prescribed
Prescribes a prioritized product backlog	Prioritization is optional
Scrum team commits to a particular amount of work for the iteration	Commitment is optional
Burndown chart is prescribed	No particular item size is prescribed
Between each sprint, a scrum board is reset	A Kanban board is persistent. It limits the number of items in workflow state
It cannot add items to ongoing iteration	It can add items whenever capacity is available
WIP limited indirectly	WIP limited directly
Timeboxed iterations prescribed	Timeboxed iterations optional

Agile metrics:

Metrics that can be collected for effective usage of Agile is:

- Drag Factor

    - Effort in hours which do not contribute to sprint goal
    - Drag factor can be improved by reducing number of shared resources, reducing the amount of non-contributing work
    - New estimates can be increased by percentage of drag factor -New estimate = (Old estimate + drag factor)
- Velocity
    - Amount of backlog (user stories) converted to shippable functionality of sprint
- No of Unit Tests added
- Time interval taken to complete daily build
- Bugs detected in an iteration or in previous iterations
- Production defect leakage

	They could look at features; e.g. iterative, incremental, modularity, adaptive/flexible, parsimony, time limits on modules/timebox, early identification/resolution of issues, client/stakeholder involvement throughout **Note:** different groups have different ideas on what the features of Agile are but the four values and twelve principles detailed below are as laid out in the Agile Manifesto. Students should understand that Agile projects should attempt to conform with the Agile Manifesto but that this may not always be possible. Students will not be required to list the values or principles in the examination. • The four Agile software development values: o Individuals and Interactions over processes and tools o Working Software over comprehensive documentation o Customer Collaboration over contract negotiation o Responding to Change over following a plan • The twelve Agile software development principles: o Customer satisfaction by early and continuous delivery of valuable software o Welcome changing requirements, even in late development o Working software is delivered frequently (weeks rather than months) o Close, daily cooperation between business people and developers o Projects are built around motivated individuals, who should be trusted o Face-to-face conversation is the best form of communication (co-location) o Working software is the primary measure of progress o Sustainable development, able to maintain a constant pace o Continuous attention to technical excellence and good design o Simplicity—the art of maximizing the amount of work not done—is essential o Best architectures, requirements, and designs emerge from self-organizing teams o Regularly, the team reflects on how to become more effective, and adjusts accordingly.
15.3.2 Agile approach phases. Understand the phases of the Agile approach, including: • requirements • plan • design • develop • release.	Guided research into the phases of the Agile approach. Students should work individually or in small groups. They should look at what happens in each phase. Groups report to whole class so that all students build up a more complete picture. **Resources** The Agile Manifesto: https://www.agilealliance.org/agile101/the-agile-manifesto/ and an introduction to agile here: https://www.youtube.com/watch?v=1iccpf2eN1Q

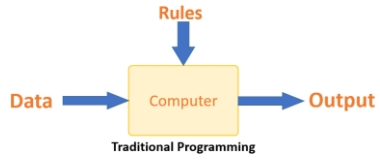
Topic 16: Emerging technologies

Machine Learning

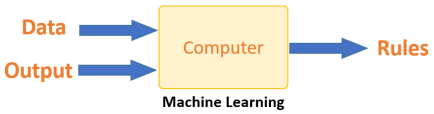
- Machine Learning is a system that can learn from example through self-improvement and without being explicitly coded by programmer. The breakthrough comes with the idea that a machine can singularly learn from the data (i.e., example) to produce accurate results.
- Machine learning **combines data** with **statistical tools** to **predict** an output. This output is then used by corporate to makes actionable insights. Machine learning is closely related to data mining and Bayesian predictive modelling. The machine receives data as input, use an algorithm to formulate answers.
- A typical machine learning tasks are to provide a recommendation. For those who have a Netflix account, all recommendations of movies or series are based on the user's historical data. Tech companies are using unsupervised learning to improve the user experience with personalizing recommendation.
- Machine learning is also used for a variety of task like fraud detection, predictive maintenance, portfolio optimization, automatize task and so on

Machine Learning vs. Traditional Programming

- In traditional programming, a programmer code all the rules in consultation with an expert in the industry for which software is being developed
- Each rule is based on a logical foundation; the machine will execute an output following the logical statement
- When the system grows complex, more rules need to be written. This can quickly become unsustainable to maintain.

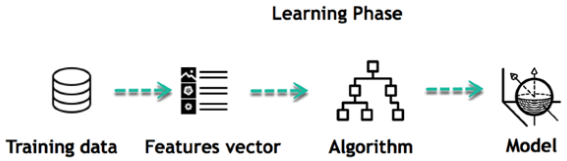


- The machine learns how the input and output data are correlated and it writes a rule
- The programmers do not need to write new rules each time there is new data
- The algorithms adapt in response to new data and experiences to improve efficacy over time.



How does Machine learning work?

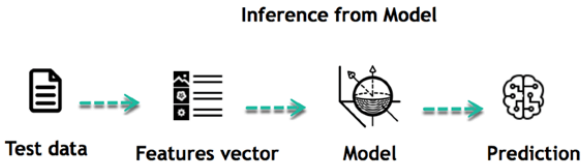
- Machine learning is the brain where all the learning takes place. The way the machine learns is similar to the human being. Humans learn from experience. The more we know, the more easily we can predict. By analogy, when we face an unknown situation, the likelihood of success is lower than the known situation.
- Machines are trained the same. To make an accurate prediction, the machine sees an example. When we give the machine a similar example, it can figure out the outcome. However, like a human, if its feed a previously unseen example, the machine has difficulties to predict.
- The core objective of machine learning is the **learning** and **inference**. First of all, the machine learns through the **discovery of patterns**. This discovery is made thanks to the data. One crucial part of the data scientist is to choose carefully which data to provide to the machine. The list of attributes used to solve a problem is called a **feature vector**. You can think of a feature vector as a subset of data that is used to tackle a problem.
- The machine uses some fancy algorithms to simplify the reality and transform this discovery into a model. Therefore, the learning stage is used to describe the data and summarize it into a model.



- For instance, the machine is trying to understand the relationship between the wage of an individual and the likelihood to go to a fancy restaurant. It turns out the machine finds a positive relationship between wage and going to a high-end restaurant: This is the model

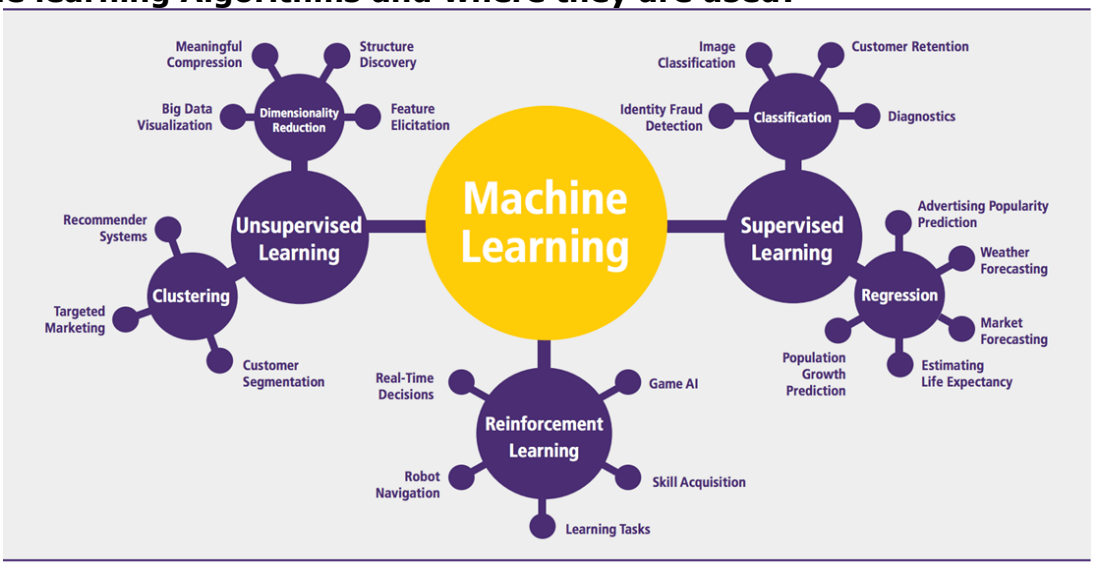
Inferring

- When the model is built, it is possible to test how powerful it is on never-seen-before data. The new data are transformed into a features vector, go through the model and give a prediction. This is all the beautiful part of machine learning. There is no need to update the rules or train again the model. You can use the model previously trained to make inference on new data.



- The life of Machine Learning programs is straightforward and can be summarized in the following points:
  1. Define a question
  2. Collect data
  3. Visualize data
  4. Train algorithm
  5. Test the Algorithm
  6. Collect feedback
  7. Refine the algorithm
  8. Loop 4-7 until the results are satisfying
  9. Use the model to make a prediction
- Once the algorithm gets good at drawing the right conclusions, it applies that knowledge to new sets of data.

Machine learning Algorithms and where they are used?



- Machine learning can be grouped into two broad learning tasks: **Supervised** and **Unsupervised**.

Supervised learning

- An algorithm uses training data and feedback from humans to learn the relationship of given inputs to a given output. For instance, a practitioner can use marketing expense and weather forecast as input data to predict the sales of cans.

- You can use supervised learning when the output data is known. The algorithm will predict new data.
- There are two categories of supervised learning:
  - **Classification task**
  - **Regression task**

1. Classification

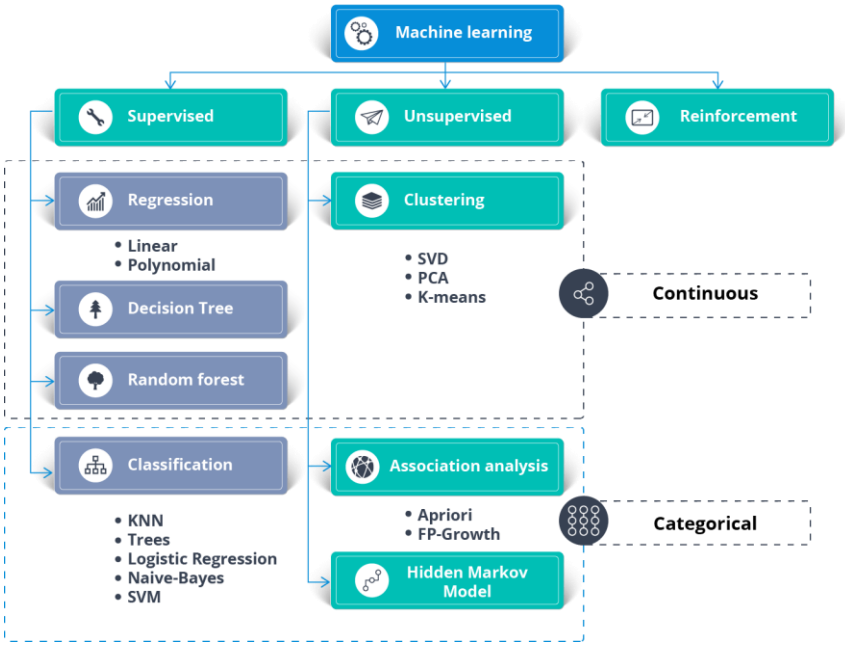
- Imagine you want to predict the gender of a customer for a commercial. You will start gathering data on the height, weight, job, salary, purchasing basket, etc. from your customer database. You know the gender of each of your customer, it can only be male or female. The objective of the classifier will be to assign a probability of being a male or a female (i.e., the label) based on the information (i.e., features you have collected). When the model learned how to recognize male or female, you can use new data to make a prediction. For instance, you just got new information from an unknown customer, and you want to know if it is a male or female. If the classifier predicts male = 70%, it means the algorithm is sure at 70% that this customer is a male, and 30% it is a female.
- The label can be of two or more classes. The above example has only two classes, but if a classifier needs to predict object, it has dozens of classes (e.g., glass, table, shoes, etc. each object represents a class)

Regression

- When the output is a continuous value, the task is a regression. For instance, a financial analyst may need to forecast the value of a stock based on a range of feature like equity, previous stock performances, macroeconomics index. The system will be trained to estimate the price of the stocks with the lowest possible error.

Algorithm used in Supervised Learning

Algorithm Name	Description	Type
Linear regression	Finds a way to correlate each feature to the output to help predict future values.	Regression
Logistic regression	Extension of linear regression that's used for classification tasks. The output variable 3is binary (e.g., only black or white) rather than continuous (e.g., an infinite list of potential colors)	Classification
Decision tree	Highly interpretable classification or regression model that splits data-feature values into branches at decision nodes (e.g., if a feature is a color, each possible color becomes a new branch) until a final decision output is made	Regression Classification
Naive Bayes	The Bayesian method is a classification method that makes use of the Bayesian theorem. The theorem updates the prior knowledge of an event with the independent probability of each feature that can affect the event.	Regression Classification
Support vector machine	Support Vector Machine, or SVM, is typically used for the classification task. SVM algorithm finds a hyperplane that optimally divided the classes. It is best used with a non-linear solver.	Regression (not very common) Classification
Random forest	The algorithm is built upon a decision tree to improve the accuracy drastically. Random forest generates many times simple decision trees and uses the 'majority vote' method to decide on which label to return. For the classification task, the final prediction will be the one with the most vote; while for the regression task, the average prediction of all the trees is the final prediction.	Regression Classification
AdaBoost	Classification or regression technique that uses a multitude of models to come up with a decision but weighs them based on their accuracy in predicting the outcome	Regression Classification
Gradient-boosting trees	Gradient-boosting trees is a state-of-the-art classification/regression technique. It is focusing on the error committed by the previous trees and tries to correct it.	Regression Classification



Unsupervised learning

- In unsupervised learning, an algorithm explores input data without being given an explicit output variable (e.g., explores customer demographic data to identify patterns)
- You can use it when you do not know how to classify the data, and you want the algorithm to find patterns and classify the data for you

Challenges and Limitations of Machine learning

- The primary challenge of machine learning is the lack of data or the diversity in the dataset. A machine cannot learn if there is no data available. Besides, a dataset with a lack of diversity gives the machine a hard time. A machine needs to have heterogeneity to learn meaningful insight. It is rare that an algorithm can extract information when there are no or few variations. It is recommended to have at least 20 observations per group to help the machine learn. This constraint leads to poor evaluation and prediction.

Application of Machine learning

Augmentation:

- Machine learning, which assists humans with their day-to-day tasks, personally or commercially without having complete control of the output. Such machine learning is used in different ways such as Virtual Assistant, Data analysis, software solutions. The primary user is to reduce errors due to human bias.

Automation:

- Machine learning, which works entirely autonomously in any field without the need for any human intervention. For example, robots performing the essential process steps in manufacturing plants.

Finance Industry

- Machine learning is growing in popularity in the finance industry. Banks are mainly using ML to find patterns inside the data but also to prevent fraud.

Government organisation

- The government makes use of ML to manage public safety and utilities. Take the example of China with the massive face recognition. The government uses Artificial intelligence to prevent jaywalker.

Healthcare industry

- Healthcare was one of the first industry to use machine learning with image detection.

Marketing

- Broad use of AI is done in marketing thanks to abundant access to data. Before the age of mass data, researchers develop advanced mathematical tools like Bayesian analysis to estimate the value of a customer. With the boom of data, marketing department relies on AI to optimize the customer relationship and marketing campaign.

Example of application of Machine Learning in Supply Chain

- Machine learning gives terrific results for visual pattern recognition, opening up many potential applications in physical inspection and maintenance across the entire supply chain network.
- Unsupervised learning can quickly search for comparable patterns in the diverse dataset. In turn, the machine can perform quality inspection throughout the logistics hub, shipment with damage and wear.
- For instance, IBM's Watson platform can determine shipping container damage. Watson combines visual and systems-based data to track, report and make recommendations in real-time.
- In past year stock manager relies extensively on the primary method to evaluate and forecast the inventory. When combining big data and machine learning, better forecasting techniques have been implemented (an improvement of 20 to 30 % over traditional forecasting tools). In term of sales, it means an increase of 2 to 3 % due to the potential reduction in inventory costs.

Example of Machine Learning Google Car

- For example, everybody knows the Google car. The car is full of lasers on the roof which are telling it where it is regarding the surrounding area. It has radar in the front, which is informing the car of the speed and motion of all the cars around it. It uses all of that data to figure out not only how to drive the car but also to figure out and predict what potential drivers around the car are going to do. What's impressive is that the car is processing almost a gigabyte a second of data.

Why is Machine Learning important?

- Machine learning is the best tool so far to analyse, understand and identify a pattern in the data. One of the main ideas behind machine learning is that the computer can be trained to automate tasks that would be exhaustive or impossible for a human being. The clear breach from the traditional analysis is that machine learning can take decisions with minimal human intervention.
- Take the following example; a retail agent can estimate the price of a house based on his own experience and his knowledge of the market.
- A machine can be trained to translate the knowledge of an expert into features. The features are all the characteristics of a house, neighbourhood, economic environment, etc. that make the price difference. For the expert, it took him probably some years to master the art of estimate the price of a house. His expertise is getting better and better after each sale.
- For the machine, it takes millions of data, (i.e., example) to master this art. At the very beginning of its learning, the machine makes a mistake, somehow like the junior salesman. Once the machine sees all the example, it got enough knowledge to make its estimation. At the same time, with incredible accuracy. The machine is also able to adjust its mistake accordingly.
- Most of the big company have understood the value of machine learning and holding data. McKinsey have estimated that the value of analytics ranges from $9.5 trillion to $15.4 trillion while $5 to 7 trillion can be attributed to the most advanced AI techniques

What is Natural Language Processing?

- Natural Language Processing (NLP) is a branch of AI that helps computers to understand, interpret and manipulate human language.
- NLP helps developers to organize and structure knowledge to perform tasks like translation, summarization, named entity recognition, relationship extraction, speech recognition, topic segmentation, etc.
- NLP is a way of computers to analyse, understand and derive meaning from a human language such as English, Spanish, Hindi, etc.

History of NLP

Here, is are important events in the history of Natural Language Processing:
1950- NLP started when Alan Turing published an article called "Machine and Intelligence."
1950- Attempts to automate translation between Russian and English
1960- The work of Chomsky and others on formal language theory and generative syntax
1990- Probabilistic and data-driven models had become quite standard
2000- A Large amount of spoken and textual data become available

How does NLP work?

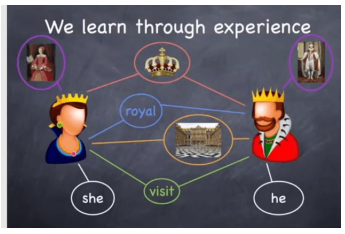
Before we learn how NLP works, let's understand how humans use language-
Every day, we say thousand of a word that other people interpret to do countless things. We, consider it as a simple communication, but we all know that words run much deeper than that. There is always some context that we derive from what we say and how we say it., NLP never focuses on voice modulation; it does draw on contextual patterns.

Example:
Man is to woman as king is to _____?
Meaning (king) – meaning (man) + meaning (woman)=?
The answer is- queen

Here, we can easily co-relate because man is male gender and woman is female gender. In the same way, the king is masculine gender, and its female gender is queen.

Example:
Is King to kings as the queen is to_____?
The answer is--- queens

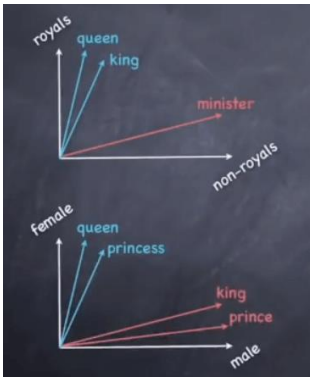
Here, we can see two words kings and kings where one is singular and other is plural. Therefore, when the world queen comes, it automatically co-relates with queens again singular plural.
Here, the biggest question is that how do we know what words mean? Let's, say who will call it queen?



The answer is we learn this thinks through experience. However, here the main question is that how computer know about the same?
We need to provide enough data for Machines to learn through experience. We can feed details like

- Her Majesty the Queen.
- The Queen's speech during the State visit
- The crown of Queen Elizabeth
- The Queens's Mother
- The queen is generous.

With above examples the machine understands the entity Queen.
The machine creates word vectors as below. A word vector is built using surrounding words.



The machine creates these vectors

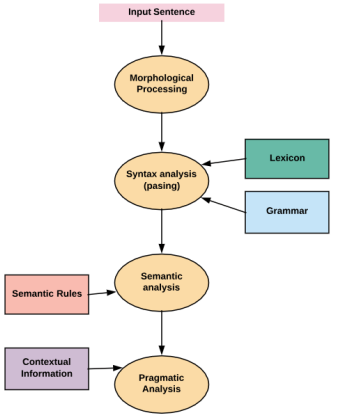
- As it learns from multiple datasets
- Use Machine learning (e.g., Deep Learning algorithms)
- A word vector is built using surrounding words.

Here is the formula:
Meaning (king) – meaning (man) + meaning (woman)=?
This amounts to performing simple algebraic operations on word vectors:
Vector (king) – vector (man) + vector (woman)= vector(?)
To which the machine answers queen.

Components of NLP

Five main Component of Natural Language processing are:

- **Morphological and Lexical Analysis**
- **Syntactic Analysis**
- **Semantic Analysis**
- **Discourse Integration**
- **Pragmatic Analysis**



Morphological and Lexical Analysis

- Lexical analysis is a vocabulary that includes its words and expressions. It depicts analysing, identifying and description of the structure of words. It includes dividing a text into paragraphs, words and the sentences
- Individual words are analysed into their components, and no word tokens such as punctuations are separated from the words.

Semantic Analysis

- Semantic Analysis is a structure created by the syntactic analyser, which assigns meanings. This component transfers linear sequences of words into structures. It shows how the words are associated with each other.
- Semantics focuses only on the literal meaning of words, phrases, and sentences. This only abstracts the dictionary meaning or the real meaning from the given context. The structures assigned by the syntactic analyser always have assigned meaning
- E.g.. "Colourless green idea." This would be rejected by the Symantec analysis as colourless here; green doesn't make any sense.

Pragmatic Analysis

- Pragmatic Analysis deals with the overall communicative and social content and its effect on interpretation. It means abstracting or deriving the meaningful use of language in situations. In this analysis, the main focus always on what was said in reinterpreted on what is meant.
- Pragmatic analysis helps users to discover this intended effect by applying a set of rules that characterize cooperative dialogues.
- E.g., "close the window?" should be interpreted as a request instead of an order.

Syntax analysis

- The words are commonly accepted as being the smallest units of syntax. The syntax refers to the principles and rules that govern the sentence structure of any individual languages.

- Syntax focus about the proper ordering of words which can affect its meaning. This involves analysis of the words in a sentence by following the grammatical structure of the sentence. The words are transformed into the structure to show hows the word are related to each other.

Discourse Integration

- It means a sense of the context. The meaning of any single sentence which depends upon that sentences. It also considers the meaning of the following sentence.
- For example, the word "that" in the sentence "He wanted that" depends upon the prior discourse context.

NLP and writing systems

The kind of writing system used for a language is one of the deciding factors in determining the best approach for text pre-processing. Writing systems can be

1. Logographic: a Large number of individual symbols represent words. Example Japanese, Mandarin
2. Syllabic: Individual symbols represent syllables
3. Alphabetic: Individual symbols represent sound

Majority of the writing systems use the Syllabic or Alphabetic system. Even English, with its relatively simple writing system based on the Roman alphabet, utilizes logographic symbols which include Arabic numerals, Currency symbols (S, £), and other special symbols.
This pose following challenges

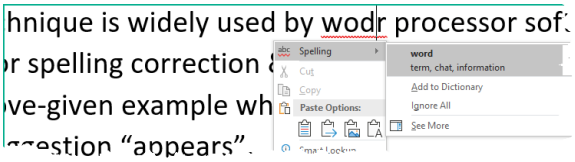
- Extracting meaning(semantics) from a text is a challenge
- NLP is dependent on the quality of the corpus. If the domain is vast, it's difficult to understand context.
- There is a dependence on the character set and language

How to implement NLP

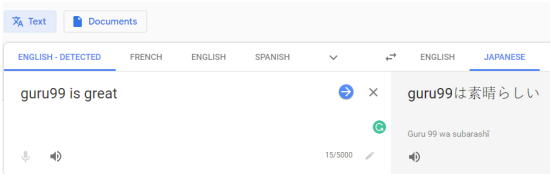
Below, given are popular methods used for Natural Learning Process:
Machine learning: The learning nlp procedures used during machine learning. It automatically focuses on the most common cases. So when we write rules by hand, it is often not correct at all concerned about human errors.
Statistical inference: NLP can make use of statistical inference algorithms. It helps you to produce models that are robust. e.g., containing words or structures which are known to everyone.

NLP Examples

Today, Natural process learning technology is widely used technology.
Here, are common Application' of NLP:
Information retrieval & Web Search
Google, Yahoo, Bing, and other search engines base their machine translation technology on NLP deep learning models. It allows algorithms to read text on a webpage, interpret its meaning and translate it to another language.
Grammar Correction:
NLP technique is widely used by word processor software like MS-word for spelling correction & grammar check.



Question Answering
Type in keywords to ask Questions in Natural Language.
Text Summarization
The process of summarising important information from a source to produce a shortened version
Machine Translation
Use of computer applications to translate text or speech from one natural language to another.



Sentiment analysis
NLP helps companies to analyse a large number of reviews on a product. It also allows their customers to give a review of the particular product.

Future of NLP

- Human readable natural language processing is the biggest Al- problem. It is all most same as solving the central artificial intelligence problem and making computers as intelligent as people.
- Future computers or machines with the help of NLP will able to learn from the information online and apply that in the real world, however, lots of work need to on this regard.
- Naturla language toolkit or nltk become more effective
- Combined with natural language generation, computers will become more capable of receiving and giving useful and resourceful information or data.

Natural language vs. Computer Language

Parameter	Natural Language	Computer Languages
Ambiguous	They are ambiguous in nature.	They are designed to unambiguous.
Redundancy	Natural languages employ lots of redundancy.	Formal languages are less redundant.
Literalness	Natural languages are made of idiom & metaphor	Formal languages mean exactly what they want to say

Advantages of NLP

- Users can ask questions about any subject and get a direct response within seconds.
- NLP system provides answers to the questions in natural language
- NLP system offers exact answers to the questions, no unnecessary or unwanted information
- The accuracy of the answers increases with the amount of relevant information provided in the question.
- NLP process helps computers communicate with humans in their language and scales other language-related tasks
- Allows you to perform more language-based data compares to a human being without fatigue and in an unbiased and consistent way.
- Structuring a highly unstructured data source

Disadvantages of NLP

- Complex Query Language- the system may not be able to provide the correct answer it the question that is poorly worded or ambiguous.
- The system is built for a single and specific task only; it is unable to adapt to new domains and problems because of limited functions.
- NLP system doesn't have a user interface which lacks features that allow users to further interact with the system

Image recognition

What is image recognition?

- Image recognition is a computer vision task that works to identify and categorize various elements of images and/or videos.
- Image recognition models are trained to take an image as input and output one or more labels describing the image.
- The set of possible output labels are referred to as **target classes**. Along with a predicted class, image recognition models may also output a confidence score related to how certain the model is that an image belongs to a class.

- For instance, if you wanted to build an image recognition model that automatically determined whether or not a dog was in a given image, the pipeline would, broadly speaking, look like this:
  - Image recognition model trained on images that have been labelled as "dog" or "not dog"
  - Model input: Image or video frame
  - Model output: Class name (i.e. dog) with a confidence score that indicates the likelihood of that image containing that class of object.

Here's what this looks like in practice:

Modes and types of image recognition

Image recognition is a broad and wide-ranging computer vision task that's related to the more general problem of pattern recognition. As such, there are a number of key distinctions that need to be made when considering what solution is best for the problem you're facing.

- Broadly speaking, we can break image recognition into two separate problems: **single** and **multiclass recognition**.
- In **single class image recognition**, models predict only one label per image. If you're training a dog or cat recognition model, a picture with a dog and a cat will still only be assigned a single label. In cases where only two classes are involved (dog; no dog), we refer to these models as **binary classifiers**.
- **Multiclass recognition models** can assign several labels to an image. An image with a cat and a dog can have one label for each. Multiclass models typically output a confidence score for each possible class, describing the probability that the image belongs to that class.

Why is image recognition important?

- Image recognition is one of the most foundational and widely-applicable computer vision tasks.
- Recognizing image patterns and extracting features is a building block of other, more complex computer vision techniques (i.e. object detection, image segmentation, etc.), but it also has numerous standalone applications that make it an essential machine learning task.
- Image recognition's broad and highly-generalizable functionality can enable a number of transformative user experiences, including but not limited to:
  - Automated image organisation
  - User-generated content moderation
  - Enhanced visual search
  - Automated photo and video tagging
  - Interactive marketing/Creative campaigns

https://www.fritz.ai/image-recognition/

Virtual Reality (VR):

- The word 'virtual' means something that is conceptual and does not exist physically and the word 'reality' means the state of being real..
- We will probably never be on the top of Mount Everest or dive deep into the Mariana Trench or step on the Moon, but we might be able to do all these things without even stepping out of our homes, this is where Virtual Reality comes to the rescue.
- In simple words **Virtual Reality means we can experience things that never actually happen. It actually tricks our brain into thinking that we are in a different place using 3 of our senses-seeing, hearing and touching.** It creates a different world, and we feel we are a part of it, both physically and mentally.

Virtual Reality (VR) has a lot of **Applications**, some of them are:

1. Entertainment-Used for gaming, 3D cinema and in theme parks.
2. Medicine-Used for surgery training, exposure therapy for people with phobia or anxiety disorder.

3. Skill training-Used for astronaut training, flight training, military training, etc.

Augmented Reality (AR):

The word '**augmented'** means to **add**.

- Augmented reality uses different tools to make the real and existing environment better and provides an improved version of reality.
- ***Snapchat filters, instagram*** *filters,* ***Pokemon*** Go are all examples of AR. It is these AR apps which allow a customer to place virtual furniture in their house before buying.
- Projection based AR applications allow human interaction by sending light onto a real surface and then sensing human touch, learning has become way more interesting and easy with AR, markerless AR provides data based on our location, it is also a powerful tool for marketing as it allows users to try products before buying.

Mixed Reality (MR):

- It brings real world and digital elements together. It integrates digital objects and real world in such a way that it makes it look like the objects really belong there.
- Mixed Reality works by scanning our physical environment and creating a map of our surroundings, so that the device will know exactly how to place digital content into that space –realistically –allowing us to interact with it.
- **Examples** of MR apps are:
  - An app that allows users to place notes around their environment.
  - A television app placed in comfortable spots for viewing.
  - A cooking app placed on the kitchen wall.
  - Microsoft's Hololens is also a famous example of MR.

Internet of Things (IoT)

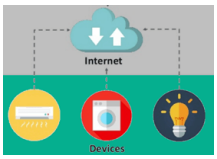
- The Internet of Things (IoT) can be defined as a network of physical objects or people called "things" that are embedded with software, electronics, network, and sensors, which allows these objects to collect and exchange data.
- The goal of IoT is to extend to internet connectivity from standard devices like computer, mobile, tablet to relatively dumb devices like a toaster
- IoT makes virtually everything "smart," by improving aspects of our life with the power of data collection, AI algorithm, and networks.
- The thing in IoT can also be a person with a diabetes monitor implant, an animal with tracking devices, etc.
-

IoT Key Features

History of IOT

- 1970- The actual idea of connected devices was proposed
- 1990- John Romkey created a toaster which could be turned on/off over the Internet
- 1995- Siemens introduced the first cellular module built for M2M
- 1999- The term "Internet of Things" was used by Kevin Ashton during his work at P&G which became widely accepted
- 2004 - The term was mentioned in famous publications like the Guardian, Boston Globe, and Scientific American
- 2005-UN's International Telecommunications Union (ITU) published its first report on this topic.
- 2008- The Internet of Things was born
- 2011- Gartner, the market research company, include "The Internet of Things" technology in their research

How IOT works?



How IoT Works

Components of an IoT system

1) Sensors/Devices: Sensors are devices that helps to collect live data from the surrounding environment. It could be a simple temperature monitoring sensor, or it may be in the form of the video feed.

2) Connectivity: All the collected data is sent to a cloud infrastructure. The sensors should be connected to the cloud using various mediums of communications. These communication mediums include mobile or satellite networks, Bluetooth, WI-FI, WAN, etc.

3) Data Processing: Once that data is collected, and it gets to the cloud, the software performs processing on the gathered data. This process can be just checking the temperature, reading on devices like AC or heaters. However, it can sometimes also be very complex like identifying objects, using computer vision on video.

4) Actuators: After data processing action need to be taken such as triggering alarms on their phones or sending them notification through email or text message. Actuators cause the user to take action

5) Storage units:

IoT Applications

IoT Applications

IoT solutions are widely used in numerous companies across industries. Some most common IoT applications are given below:

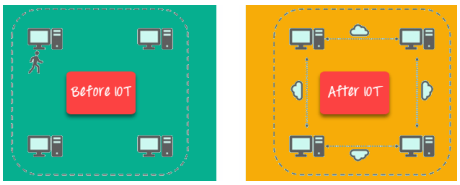
Application type	Description
Smart Thermostats	Helps you to save resource on heating bills by knowing your usage patterns.
Connected Cars	IOT helps automobile companies handle billing, parking, insurance, and other related stuff automatically.
Activity Trackers	Helps you to capture heart rate pattern, calorie expenditure, activity levels, and skin temperature on your wrist.
Smart Outlets	Remotely turn any device on or off. It also allows you to track a device's energy level and get custom notifications directly into your smartphone.
Parking Sensors	IOT technology helps users to identify the real-time availability of parking spaces on their phone.
Connect Health	The concept of a connected health care system facilitates real-time health monitoring and patient care. It helps in improved medical decision-making based on patient data.
Smart City	Smart city offers all types of use cases which include traffic management to water distribution, waste management, etc.
Smart home	Smart home encapsulates the connectivity inside your homes. It includes smoke detectors, home appliances, light bulbs, windows, door locks, etc.
Smart supply chain	Helps you in real time tracking of goods while they are on the road, or getting suppliers to exchange inventory information.

Challenges of IoT

At present IoT is faced with many challenges, such as:

- Insufficient testing and updating
- Concern regarding data security and privacy
- Software complexity
- Data volumes and interpretation
- Integration with AI and automation
- Devices require a constant power supply which is difficult
- Interaction and short-range communication

Advantages of IoT



- **Technical Optimization:** IoT technology helps a lot in improving technologies and making them better. Example, with IoT, a manufacturer is able to collect data from various car sensors. The manufacturer analyses them to improve its design and make them more efficient.
- **Improved Data Collection:** Traditional data collection has its limitations and its design for passive use. IoT facilitates immediate action on data.
- **Reduced Waste:** IoT offers real-time information leading to effective decision making & management of resources. For example, if a manufacturer finds an issue in multiple car engines, he can track the manufacturing plan of those engines and solves this issue with the manufacturing belt.

- **Improved Customer Engagement:** IoT allows you to improve customer experience by detecting problems and improving the process.

Disadvantages IOT

- **Security:** IoT technology creates an ecosystem of connected devices. However, during this process, the system may offer little authentication control despite sufficient security measures.
- **Privacy:** The use of IOT, exposes a substantial amount of personal data, in extreme detail, without the user's active participation. This creates lots of privacy issues.
- **Flexibility:** There is a huge concern regarding the flexibility of an IoT system. It is mainly regarding integrating with another system as there are many diverse systems involved in the process.
- **Complexity:** The design of the IOT system is also quite complicated. Moreover, it's deployment and maintenance also not very easy.
- **Compliance:** IOT has its own set of rules and regulations. However, because of its complexity, the task of compliance is quite challenging.

Internet of Things - Hardware

The hardware utilized in IoT systems includes devices for a remote dashboard, devices for control, servers, a routing or bridge device, and sensors. These devices manage key tasks and functions such as system activation, action specifications, security, communication, and detection to support-specific goals and actions.

IoT – Sensors

The most important hardware in IoT might be its sensors. These devices consist of energy modules, power management modules, RF modules, and sensing modules. RF modules manage communications through their signal processing, WiFi, ZigBee, Bluetooth, radio transceiver, duplexer, and BAW.

The sensing module manages sensing through assorted active and passive measurement devices. Here is a list of some of the measurement devices used in IoT –

Devices	
accelerometers	temperature sensors
magnetometers	proximity sensors
gyroscopes	image sensors
acoustic sensors	light sensors
pressure sensors	gas RFID sensors
humidity sensors	micro flow sensors

Wearable Electronics

Wearable electronic devices are small devices worn on the head, neck, arms, torso, and feet.

Smartwatches not only help us stay connected, but as a part of an IoT system, they allow access needed for improved productivity.

Current smart wearable devices include –

- **Head** – Helmets, glasses
- **Neck** – Jewelry, collars
- **Arm** – Watches, wristbands, rings
- **Torso** – Clothing, backpacks
- **Feet** – Socks, shoes

Smart glasses help us enjoy more of the media and services we value, and when part of an IoT system, they allow a new approach to productivity.

Standard Devices

The desktop, tablet, and cell phone remain integral parts of IoT as the command centre and remotes.

- The **desktop** provides the user with the highest level of control over the system and its settings.
- The **tablet** provides access to the key features of the system in a way resembling the desktop, and also acts as a remote.
- The **cell phone** allows some essential settings modification and also provides remote functionality.

https://www.tutorialspoint.com/internet_of_things/index.htm

IoT standards and frameworks

There are several emerging IoT standards, including:

- 6LoWPAN (IPv6 over Low -Power Wireless Personal Area Networks), an open standard defined by the Internet Engineering Task Force (IETF). The 6LoWPAN standard enables any low-power radio to communicate to the internet, including 804.15.4, Bluetooth Low Energy and Z-Wave (for home automation).
- ZigBee0, a low-power, low data-rate wireless network used mainly in industrial settings. ZigBee is based on based the IEEE 802.15.4 standard. The ZigBee Alliance created Dotdot, the universal language for IoT that enables smart objects to work securely on any network and understand each other.
- LiteOS, a Unix-like operating system for wireless sensor networks. LiteOS supports smartphones, wearables, intelligent manufacturing applications, smart homes and Internet of Vehicles (IoV). The operating system also serves as a smart device development platform.
- OneM2M, a machine-to-machine service layer that can be embedded in software and hardware to connect devices. The global standardisation body, OneM2M, was created to develop reusable standards to enable IoT applications across different verticals to communicate.
- DDS (Data Distribution Service) was developed by the Object Management Group (OMG) and is an IoT standard for real-time, scalable and high-performance machine-to-machine communication.
- AMQP (Advanced Message Queuing Protocol), an open source published standard for asynchronous messaging by wire. AMQP enables encrypted and interoperable messaging between organisations and applications. The protocol is used in client/server messaging and in IoT device management.
- CoAP (Constrained Application Protocol), a protocol designed by the IETF that specifies how low-power compute-constrained devices can operate in the internet of things.
- LoRaWAN (Long Range Wide Area Network), a protocol for wide area networks, it's designed to support huge networks, such as smart cities, with millions of low-power devices.

IoT frameworks include:

- AWS IoT, a cloud platform for IoT released by Amazon. This framework is designed to enable smart devices to easily connect and securely interact with the AWS cloud and other connected devices.
- ARM Mbed IoT, a platform to develop apps for the IoT based on ARM microcontrollers. The goal of the ARM Mbed IoT platform is to provide a scalable, connected and secure environment for IoT devices by integrating Mbed tools and services.
- Microsoft's Azure IoT Suite, a platform that consists of a set of services that enables users to interact with and receive data from their IoT devices as well as perform various operations over data, such as multidimensional analysis, transformation and aggregation, and visualise those operations in a way that's suitable for business.
- Google's Brillo/Weave, a platform for the rapid implementation of IoT applications. The platform consists of two main backbones: Brillo, an android-based operating system for the development of embedded low power devices; and Weave, IoT-oriented communication protocol that serves as the communication language between the device and the cloud.
- Calvin, an open source IoT platform released by Ericsson designed for building and managing distributed applications that enable devices talk to each other. Calvin includes a development framework for application developers as well as a runtime environment for handling the running application.

IoT security and privacy issues

- The internet of things connects billions of devices to the internet and involves the use of billions of data points, all of which need to be secured. Due to its expanded attack surface, IoT security and IoT privacy are cited as major concerns.
- In 2016, one of the most notorious recent IoT attacks was Mirai, a botnet that infiltrated domain name server provider Dyn and took down many websites for an extended period of time in one of the biggest distributed denial-of-service (DDoS)

attacks ever seen. Attackers gained access to the network by exploiting poorly secured IoT devices.

- Because IoT devices are closely connected, all a hacker has to do is exploit one vulnerability to manipulate all the data, rendering it unusable. Manufacturers that don't update their devices regularly -- or at all -- leave them vulnerable to cybercriminals.
- Additionally, connected devices often ask users to input their personal information, including names, ages, addresses, phone numbers and even social media accounts -- information that's invaluable to hackers.
- However, hackers aren't the only threat to the internet of things; privacy is another major concern for IoT users. For instance, companies that make and distribute consumer IoT devices could use those devices to obtain and sell users' personal data.
- Beyond leaking personal data, IoT poses a risk to critical infrastructure, including electricity, transportation and financial services.

The future of IoT

There is no shortage of IoT market estimations. For example, a few include:

- Bain & Company expects annual IoT revenue of hardware and software to exceed $450 billion by 2020.
- McKinsey & Company estimates IoT will have an $11.1 trillion impact by 2025.
- IHS Markit believes the number of connected IoT devices will increase 12% annually to reach 125 billion in 2030.
- Gartner assesses that 20.8 billion connected things will be in use by 2020, with total spend on IoT devices and services to reach $3.7 trillion in 2018.